Computer science involves the application of theoretical concepts in the context of software development to the solution of problems that arise in almost every human endeavor. Computer science as a discipline draws its inspiration from mathematics, logic, science, and engineering. From these roots, computer science has fashioned paradigms for program structures, algorithms, data representations, efficient use of computational resources, robustness and security, and communication within computers and across networks. The ability to frame problems, select computational models, design program structures, and develop efficient algorithms is as important in computer science as software implementation skill. Computer science is concerned with bringing together all of the intellectual resources needed to enable the rapid and effective development of software to meet the needs of business, research, and end users.

The goal of the undergraduate program in computer science is to teach students the conceptual and practical skills that will enable them to contribute to the development of computational principles and to play a productive role in the software community. To that end, the undergraduate program focuses on the fundamentals of program design including object-oriented design, software development, computer organization, systems and networks, theory of computation, principles of languages, and advanced algorithms and data. The program also offers a variety of electives at the upper undergraduate and beginning graduate levels ranging from more theoretical courses to those that focus on important applications.

The Bachelor of Science in Computer Science with Concentration in Cyber Operations is one of the initial four programs selected in 2012 by the National Security Agency as a National Center of Academic Excellence in Cyber Operations Program.

**Programs**

**Bachelor of Science in Computer Science (BSCS)**
- Computer Science (http://catalog.northeastern.edu/undergraduate/computer-information-science/computer-science/bscs)
- Computer Science with Concentration in Cyber Operations (http://catalog.northeastern.edu/undergraduate/computer-information-science/computer-science/concentration-cyber-operations-bscs)

**Bachelor of Arts in Computer Science (BACS)**
- Computer Science (http://catalog.northeastern.edu/undergraduate/computer-information-science/computer-science/bacs)

**Bachelor of Science (BS)**
- Cybersecurity (http://catalog.northeastern.edu/undergraduate/computer-information-science/computer-science/cyber-security-bs)

**Minor**
- Computer Science (http://catalog.northeastern.edu/undergraduate/computer-information-science/computer-science/minor)

**Accelerated Programs**
See Accelerated Bachelor/Graduate Degree Programs (http://catalog.northeastern.edu/undergraduate/computer-information-science/accelerated-bachelor-graduate-degree-programs/#programtext)

**Courses**

**Computer Science Courses**

**CS 1100. Computer Science and Its Applications. 4 Hours.**
Introduces students to the field of computer science and the patterns of thinking that enable them to become intelligent users of software tools in a problem-solving setting. Examines several important software applications so that students may develop the skills necessary to use computers effectively in their own disciplines.

**CS 1200. Computer Science/Information Science Overview 1. 1 Hour.**
Introduces students to the College of Computer and Information Science (CCIS) and begins their preparation for careers in the computing and information fields. Offers students an opportunity to learn how to thrive at Northeastern and within CCIS by developing academic, professional, and interpersonal skills. Covers the variety of careers available in the high-technology professions. Students work in groups to create and deliver presentations on careers in the field. Intended for freshmen.

**CS 1210. Computer Science/Information Science Overview 2: Co-op Preparation. 1 Hour.**
Continues the preparation of students for careers in the computing and information fields by discussing co-op and co-op processes. Offers students an opportunity to prepare a professional résumé; practice proper interviewing techniques; explore current job opportunities; learn how to engage in the job and referral process; and to understand co-op policies, procedures, and expectations. Discusses professional behavior and ethical issues in the workplace.

**CS 1220. Computer/Information Science Co-op Preparation. 1 Hour.**
Prepares students for co-op through topics such as ethics, privacy, security, responsibility, and intellectual property. Exposes students to popular industry technologies. Intended for students who transfer into computer/information science who are above the freshman level.

**CS 1500. Algorithms and Data Structures for Engineering. 4 Hours.**
Introduces algorithms and data structures for engineering students. Discusses data structures such as arrays, stacks, queues, and lists, and the algorithms that manipulate these structures. Introduces simple algorithm analysis. Discusses classes and objects and presents the basic material about encapsulation, inheritance, and polymorphism. Introduces software development practices such as modular design, use of libraries, testing methods, and debugging techniques.

**CS 1501. Lab for CS 1500. 1 Hour.**
Accompanies CS 1500. Covers topics from the course through various experiments.

**CS 1800. Discrete Structures. 4 Hours.**
Introduces the mathematical structures and methods that form the foundation of computer science. Studies structures such as sets, tuples, sequences, lists, trees, and graphs. Discusses functions, relations, ordering, and equivalence relations. Examines inductive and recursive definitions of structures and functions. Discusses principles of proof such as truth tables, inductive proof, and basic logic. Also covers the counting techniques and arguments needed to estimate the size of sets, the growth of functions, and the space-time complexity of algorithms.

**CS 1801. Recitation for CS 1800. 0 Hours.**
Accompanies CS 1800. Provides students with additional opportunities to ask questions and to see sample problems solved in detail.
CS 2800. Logic and Computation. 4 Hours.
Introduces formal logic and its connections to computer and information science. Offers an opportunity to learn to translate statements about the behavior of computer programs into logical claims and to gain the ability to prove such assertions both by hand and using automated tools. Considers approaches to proving termination, correctness, and safety for programs. Discusses notations used in logic, propositional and first order logic, logical inference, mathematical induction, and structural induction. Introduces the use of logic for modeling the range of artifacts and phenomena that arise in computer and information science.

CS 2801. Lab for CS 2800. 1 Hour.
Accompanies CS 2800. Covers topics from the course through various experiments.

CS 2990. Elective. 1-4 Hours.
Offers elective credit for courses taken at other academic institutions. May be repeated without limit.

CS 3200. Database Design. 4 Hours.
Studies the design of a database for use in a relational database management system. The entity-relationship model and normalization are used in problems. Relational algebra and then the SQL (structured query language) are presented. Advanced topics include triggers, stored procedures, indexing, elementary query optimization, and fundamentals of concurrency and recovery. Students implement a database schema and short application programs on one or more commercial relational database management systems.

CS 3500. Object-Oriented Design. 4 Hours.
Presents a comparative approach to object-oriented programming and design. Discusses the concepts of object, class, meta-class, message, method, inheritance, and genericity. Reviews forms of polymorphism in object-oriented languages. Contrasts the use of inheritance and composition as dual techniques for software reuse: forwarding vs. delegation and subclassing vs. subtyping. Fosters a deeper understanding of the principles of object-oriented programming and design including software components, object-oriented design patterns, and the use of graphical design notations such as UML (unified modeling language). Basic concepts in object-oriented design are illustrated with case studies in application frameworks and by writing programs in one or more object-oriented languages.

CS 3520. Programming in C++. 4 Hours.
Examines how to program in C++ in a robust and safe manner. Reviews basics, including scoping, typing, and primitive data structures. Discusses data types (primitive, array, structure, class, string); addressing/parameter mechanisms (value, pointer, reference); stacks; queues; linked lists; binary trees; hash tables; and the design of classes and class inheritance, emphasizing single inheritance. Considers the instantiation of objects, the trade-offs of stack vs. heap allocation, and the design of constructors and destructors. Emphasizes the need for a strategy for dynamic memory management. Addresses function and operator overloading; templates, the Standard Template Library (STL), and the STL components (containers, generic algorithms, iterators, adaptors, allocators, function objects); streams; exception handling; and system calls for processes and threads.

CS 3540. Game Programming. 4 Hours.
Introduces the different subsystems used to create a 3D game, including rendering, animation, collision, physics, audio, trigger systems, game logic, behavior trees, and simple artificial intelligence. Offers students an opportunity to learn the inner workings of game engines and how to use multiple libraries such as physics and graphics libraries to develop a game. Discusses graphics pipeline, scene graph, level design, behavior scripting, object-oriented game design, world editors, and game scripting languages.
CS 3600. Systems and Networks. 4 Hours.
Introduces the basic concepts underlying computer operating systems and computer networks and provides hands-on experience with their implementation. Covers the basic structure of an operating system: application interfaces, processes, threads, synchronization, interprocess communication, processor allocation, deadlocks, memory management, file systems, and input/output control. Also introduces network architectures, network topologies, network protocols, layering concepts (for example, ISO/OSI, TCP/IP reference models), communication paradigms (point-to-point vs. multicast/broadcast, connectionless vs. connection oriented), and networking API's (sockets). Uses examples from many real operating systems and networks (UNIX, MS-DOS, Windows, TCP/IP, Ethernet, ATM, and token rings) to reinforce concepts.

CS 3650. Computer Systems. 4 Hours.
Introduces the basic design of computing systems, computer operating systems, and assembly language using a RISC architecture. Describes caches and virtual memory. Covers the interface between assembly language and high-level languages, including call frames and pointers. Covers the use of system calls and systems programming to show the interaction with the operating system. Covers the basic structures of an operating system, including application interfaces, processes, threads, synchronization, interprocess communication, deadlock, memory management, file systems, and input/output control.

CS 3700. Networks and Distributed Systems. 4 Hours.
Introduces the fundamentals of computer networks, including network architectures, network topologies, network protocols, layering concepts (for example, ISO/OSI, TCP/IP reference models), communication paradigms (point-to-point vs. multicast/broadcast, connectionless vs. connection oriented), and networking APIs (sockets). Also covers the construction of distributed programs, with an emphasis on high-level protocols and distributed state sharing. Topics include design patterns, transactions, performance trade-offs, security implications, and reliability. Uses examples from real networks (TCP/IP, Ethernet, 802.11) and distributed systems (Web, BitTorrent, DNS) to reinforce concepts.

CS 3740. Systems Security. 4 Hours.
Introduces the fundamental principles of designing and implementing secure programs and systems. Presents and analyzes prevalent classes of attacks against systems. Discusses techniques for identifying the presence of vulnerabilities in system design and implementation, preventing the introduction of or successful completion of attacks, limiting the damage incurred by attacks, and strategies for recovering from system compromises. Offers opportunities for hands-on practice of real-world attack and defense in several domains, including systems administration, the Web, and mobile devices. Presents the ethical considerations of security research and practice.

CS 3800. Theory of Computation. 4 Hours.
Introduces the theory behind computers and computing aimed at answering the question, “What are the capabilities and limitations of computers?” Covers automata theory, computability, and complexity. The automata theory portion includes finite automata, regular expressions, nondeterminism, nonregular languages, context-free languages, pushdown automata, and noncontext-free languages. The computability portion includes Turing machines, the Church-Turing thesis, decidable languages, and the Halting theorem. The complexity portion includes big-O and small-o notation, the classes P and NP, the P vs. NP question, and NP-completeness.

CS 3990. Elective. 1-4 Hours.
Offers elective credit for courses taken at other academic institutions. May be repeated without limit.

CS 4000. Senior Seminar. 1 Hour.
Requires students to give a twenty- to thirty-minute formal presentation on a topic of their choice in computer science. Prepares students for this talk by discussing methods of oral presentation, how to present technical material, how to choose what topics to present, overall organization of a talk, and use of presentation software and other visual aids.

CS 4100. Artificial Intelligence. 4 Hours.
Introduces the fundamental problems, theories, and algorithms of the artificial intelligence field. Includes heuristic search; knowledge representation using predicate calculus; automated deduction and its applications; planning; and machine learning. Additional topics include game playing; uncertain reasoning and expert systems; natural language processing; logic for common-sense reasoning; ontologies; and multiagent systems.

CS 4120. Natural Language Processing. 4 Hours.
Introduces the computational modeling of human language; the ongoing effort to create computer programs that can communicate with people in natural language; and current applications of the natural language field, such as automated document classification, intelligent query processing, and information extraction. Topics include computational models of grammar and automatic parsing, statistical language models and the analysis of large text corpora, natural language semantics and programs that understand language, models of discourse structure, and language use by intelligent agents. Course work includes formal and mathematical analysis of language models and implementation of working programs that analyze and interpret natural language text. Knowledge of statistics is helpful.

CS 4150. Game Artificial Intelligence. 4 Hours.
Offers an overview of classical and modern approaches to artificial intelligence in digital games. Focuses on the creation of believable agents and environments with the goal of providing a fun and engaging experience to a player. Covers player modeling, procedural content generation, behavior trees, interactive narrative, decision-making systems, cognitive modeling, and path planning. Explores different approaches for behavior generation, including learning and rule-based systems. Requires students to complete several individual assignments in these areas to apply the concepts covered in class. Students choose a group final project to explore one aspect of artificial intelligence for games in further depth. Offers students an opportunity to learn team management and communication. Students who do not meet course prerequisites may seek permission of instructor.

CS 4200. Database Internals. 4 Hours.
Explores the internal workings of database management systems. Explains how database systems store data on disks. Studies how to improve query efficiency using index techniques such as B+-tree, hash indices, and multidimensional indices. Describes how queries are executed internally and how database systems perform query optimizations. Introduces concurrency control schemes implemented by locking, such as hierarchical locking and key range locking. Discusses lock table structure. Discusses how database systems can perform logging and recovery to avoid loss of data in case of system crashes.
CS 4240. Large-Scale Parallel Data Processing. 4 Hours.
Covers techniques for managing and analyzing very large data sets, with an emphasis on approaches that scale out effectively as more compute nodes are added. Introduces principles of distributed data management and strategies for problem-driven data partitioning through a selection of design patterns from various application domains, including graph analysis, databases, text processing, and data mining. Offers students an opportunity to obtain hands-on programming experience with modern big-data processing technology such as MapReduce, Spark, HBase, and cloud computing (this selection is subject to change as technology evolves).

CS 4300. Computer Graphics. 4 Hours.
Charts a path through every major aspect of computer graphics with varying degrees of emphasis. Discusses hardware issues: size and speed; lines, polygons, and regions; modeling, or objects and their relations; viewing, or what can be seen (visibility and perspective); rendering, or how it looks (properties of surfaces, light, and color); transformations, or moving, placing, distorting, and animating and interaction, or drawing, selecting, and transforming.

CS 4400. Programming Languages. 4 Hours.
Introduces a systematic approach to understanding the behavior of programming languages. Covers interpreters; static and dynamic scope; environments; binding and assignment; functions and recursion; parameter-passing and method dispatch; objects, classes, inheritance, and polymorphism; type rules and type checking; and concurrency.

CS 4410. Compilers. 4 Hours.
Studies the construction of compilers and integrates material from earlier courses on programming languages, automata theory, computer architecture, and software design. Examines syntax trees; static semantics; type checking; typical machine architectures and their software structures; code generation; lexical analysis; and parsing techniques. Uses a hands-on approach with a substantial term project.

CS 4500. Software Development. 4 Hours.
Considers software development as a systematic process involving specification, design, documentation, implementation, testing, and maintenance. Examines software process models; methods for software specification; modularity, abstraction, and software reuse; and issues of software quality. Students, possibly working in groups, design, document, implement, test, and modify software projects.

CS 4510. Software Testing. 4 Hours.
Examines the software development process from the point of view of testing. Focuses on unit testing, white- and black-box testing, randomized testing, the design of equality comparison, and the design of a test tool that evaluates the tests and reports the results. Next considers integration testing, stress tests and other performance tests, testing automation, and other techniques for assuring correctness and integrity of programs with several interacting components. Explores tools for measuring code quality and how these tools may be used to improve code design. Requires a comprehensive project in which all techniques studied are applied.

CS 4520. Mobile Application Development. 4 Hours.
Focuses on mobile application development on a mobile phone or related platform. Discusses memory management; user interface building, including both MVC principles and specific tools; touch events; data handling, including core data, SQL, XML, and JSON; network techniques and URL loading; and, finally, specifics such as GPS and motion sensing that may be dependent on the particular mobile platform. Students are expected to work on a project that produces a professional-quality mobile application. The instructor chooses a modern mobile platform to be used in the course.

CS 4550. Web Development. 4 Hours.
Discusses Web development for sites that are dynamic, data driven, and interactive. Focuses on the software development issues of integrating multiple languages, assorted data technologies, and Web interaction. Considers ASP.NET, C#, HTTP, HTML, CSS, XML, XSLT, JavaScript, AJAX, RSS/Atom, SQL, and Web services. Requires each student to deploy individually designed Web experiments that illustrate the Web technologies and at least one major integrative Web site project. Students may work as a team with the permission of the instructor. Each student or team must also create extensive documentation of their goals, plans, design decisions, accomplishments, and user guidelines. All source files must be open and be automatically served by a sources server.

CS 4600. Topics in Operating Systems. 4 Hours.
Studies advanced concepts underlying computer operating systems and computer networks. Examines in depth all major operating-system and network components including device drivers, network protocol stacks, memory managers, centralized and distributed file systems, interprocess communication mechanisms, real-time schedulers, and security mechanisms. Additional components are covered as time permits. Provides hands-on experience with the source code of commercial-grade operating systems and networks.

CS 4610. Robotic Science and Systems. 4 Hours.
Introduces autonomous mobile robots, with a focus on algorithms and software development, including closed-loop control, robot software architecture, wheeled locomotion and navigation, tactile and basic visual sensing, obstacle detection and avoidance, and grasping and manipulation of objects. Offers students an opportunity to progressively construct mobile robots from a predesigned electromechanical kit. The robots are controlled wirelessly by software of the students’ own design, built within a provided robotics software framework. The course culminates in a grand challenge competition using all features of the robots.

CS 4611. Lab for CS 4610. 1 Hour.
Offers a laboratory course to accompany CS 4610.

CS 4620. Building Extensible Systems. 4 Hours.
Deals with the design of extensible software systems, which enable clients to add functionality both statically as well as dynamically. Examples of such systems are operating systems, game servers, and Web browsers. Describes the classic systems built on C-like languages with unsafe, manual memory control and the more recent systems built on Java-like languages with safe, automated memory management. Introduces the Rust programming language, which combines the efficiency of C with safe manual memory control via type specifications and compiler constraints. Offers students an opportunity to build systems using all three settings but focuses on the Rust approach. Students also have an opportunity to evaluate their work via essays and memos.

CS 4650. High Performance Computing. 4 Hours.
Introduces students to research in the domain of high-performance computing. Each instance of this course covers a single topic with broad open questions. The required systems background needed to investigate these questions is covered in the first part of the course. Then, working in teams, students have an opportunity to address different aspects of the open questions so that in combination the entire class may learn more than any single team could accomplish. Example topics include use of new hardware such as GPUs on video boards, use of new software tools for multicore computing, development of check-pointing packages for more robust long computations, software for GUI window systems, and cloud computing. May be repeated once.
CS 4700. Network Fundamentals. 4 Hours.
Introduces the fundamental concepts of network protocols and network architectures. Presents the different harmonizing functions needed for the communication and effective operation of computer networks. Provides in-depth coverage of data link control, medium access control, routing, end-to-end transport protocols, congestion and flow control, multicasting, naming, auto configuration, quality of service, and network management. Studies the abstract mechanisms and algorithms as implemented in real-world Internet protocols. Also covers the most common application protocols (e-mail, Web, and ftp).

CS 4710. Mobile and Wireless Systems. 4 Hours.
Covers both theoretical foundations of wireless/mobile networking and practical aspects of wireless/mobile systems, including current standards, mobile development platforms, and emerging technologies. Incorporates a strong practical component; requires students to work in teams on several practical assignments (e.g., based on Wi-Fi sensing, mobile applications, Internet-of-Things devices, and software-defined radio applications) and a final project. The final project integrates knowledge about several wireless communication technologies and mechanisms.

CS 4740. Network Security. 4 Hours.
Studies topics related to Internet architecture and cryptographic schemes in the context of security. Provides advanced coverage of the major Internet protocols including IP and DNS. Examines denial of service, viruses, and worms, and discusses techniques for protection. Covers cryptographic paradigms and algorithms such as RSA and Diffie-Hellman in sufficient mathematical detail. The advanced topics address the design and implementation of authentication protocols and existing standardized security protocols. Explores the security of commonly used applications like the Web and e-mail.

CS 4750. Secure Wireless Ad Hoc Robots on Mission (SWARM) 1. 4 Hours.
Introduces the concepts underlying the design of robust and secure heterogeneous wireless networking of mobile robots: Internetworking, security, wireless communication, embedded development, and mobile phone platforms. Students form mixed teams with the goal of designing and building rescue-mission-oriented heterogeneous wireless systems operating in adversarial environments. These systems consist of off-the-shelf robots enhanced by the students with a low-power control and sensing embedded system; a low-power digital radio frequency communication network; a coordination unit connected to the Internet; and a messaging and command system based on cell phones. The course culminates in a competition between teams. Students are graded based on their designs, presentations, innovation, robustness, and competition performance. Graduate students are expected to make a research contribution.

CS 4760. Secure Wireless Ad Hoc Robots on Mission (SWARM) 2. 4 Hours.
Continues CS 4750. Based on the experiences in CS 4750, student teams have an opportunity to build more autonomous systems that can navigate areas where wireless communication or direct visibility are not possible. The systems must be resilient to more sophisticated denial-of-service attacks and need to more carefully account for energy consumption expended on mobility, communication, and meeting the mission task. Graduate students are expected to make a research contribution.

CS 4770. Cryptography. 4 Hours.
Studies the design of cryptographic schemes that enable secure communication and computation. Emphasizes cryptography as a mathematically rigorous discipline with precise definitions, theorems, and proofs and highlights deep connections to information theory, computational complexity, and number theory. Topics include pseudorandomness; symmetric-key cryptosystems and block ciphers such as AES; hash functions; public-key cryptosystems, including ones based on factoring and discrete logarithms; signature schemes; secure multiparty computation and applications such as auctions and voting; and zero-knowledge proofs.

CS 4800. Algorithms and Data. 4 Hours.
Introduces the basic principles and techniques for the design, analysis, and implementation of efficient algorithms and data representations. Discusses asymptotic analysis and formal methods for establishing the correctness of algorithms. Considers divide-and-conquer algorithms, graph traversal algorithms, and optimization techniques. Introduces information theory and covers the fundamental structures for representing data. Examines flat and hierarchical representations, dynamic data representations, and data compression. Concludes with a discussion of the relationship of the topics in this course to complexity theory and the notion of the hardness of problems.

CS 4850. Building Game Engines. 4 Hours.
Discusses the components of game engines and strategies for their software implementation. Includes graphics management algorithms (animation, scene graph, level of detail); basic artificial intelligence algorithms (search, decision making, sensing); and related algorithmic issues (networking, threading, input processing). Explores the use of data-driven software design. Offers students an opportunity to use a rendering engine and to build and integrate several software components to create a complete game engine. Requires students to work on several individual assignments to apply the algorithms and then develop a project in a team. Offers students an opportunity to learn team/project management; work division; team communication; and the software development cycle of implementation, testing, critique, and further iteration. Students who do not meet course prerequisites may seek permission of instructor.

CS 4900. Honors Senior Seminar. 4 Hours.
Offers a capstone course for computer science honors students. Exposes students to one or more topics of current interest in computer science. Requires students to prepare a one-hour presentation on a topic in computer science and to write a paper on that topic.

CS 4910. Computer Science Topics. 4 Hours.
Offers a lecture course in computer science on a topic not regularly taught in a formal course. Topics may vary from offering to offering. May be repeated up to three times.

CS 4920. Computer Science Project. 4 Hours.
Focuses on students developing a substantial software or hardware artifact under faculty supervision. May be repeated up to three times.

CS 4930. Cybersecurity Capstone. 4 Hours.
Provides the culmination of the learned principles and methodologies for identifying and addressing cybersecurity issues in organizations. Offers students an opportunity to work in small groups to identify and scope a current cybersecurity problem/challenge. Requires students to submit a written proposal about the project, complete with motivation, literature research, and reasons for the study; create a work plan to develop a solution to include the development and identification of the data necessary to properly solve the problem/challenge; and create a final report.
CS 4940. Research Projects on National Security. 4 Hours.
Engages students in national cybersecurity/information systems security problems. Offers students an opportunity to learn how to apply research techniques, think clearly about these issues, formulate and analyze potential solutions, and communicate their results. Working in small groups under the mentorship of external mentors from government and industry, each student has an opportunity to formulate, carry out, and present original research on current cybersecurity/information assurance problems of interest to the nation. As part of this research, students are required to submit a written proposal about the project, complete with motivation, literature research, and reasons for the study; create a work plan for the research problem; and create a final report.

CS 4970. Junior/Senior Honors Project 1. 4 Hours.
Focuses on second semester of in-depth project in which a student conducts research or produces a product related to the student’s major field. Combined with Junior/Senior Project 2 or college-defined equivalent for 8 credit honors project. May be repeated without limit.

CS 4971. Junior/Senior Honors Project 2. 4 Hours.
Focuses on second semester of in-depth project in which a student conducts research or produces a product related to the student’s major field. May be repeated without limit.

CS 4990. Elective. 1-4 Hours.
Offers elective credit for courses taken at other academic institutions. May be repeated without limit.

CS 4991. Research. 4,8 Hours.
Offers an opportunity to conduct research under faculty supervision. May be repeated up to three times.

CS 4992. Directed Study. 1-6 Hours.
Focuses on student examining standard computer science material in fresh ways or new computer science material that is not covered in formal courses. May be repeated up to three times.

CS 4993. Independent Study. 1-6 Hours.
Offers independent work under the direction of members of the department on a chosen topic. Course content depends on instructor. May be repeated up to three times.

CS 4994. Internship. 4 Hours.
Offers students an opportunity for internship work. May be repeated without limit.

CS 4996. Experiential Education Directed Study. 4 Hours.
Draws upon the student’s approved experiential activity and integrates it with study in the academic major. Restricted to those students who are using the course to fulfill their experiential education requirement.

CS 4997. Computer Science Thesis. 4 Hours.
Focuses on student preparing an undergraduate thesis under faculty supervision.

CS 4998. Computer Science Thesis Continuation. 4 Hours.
Focuses on student continuing to prepare an undergraduate thesis under faculty supervision.

CS 5001. Intensive Foundations of Computer Science. 4 Hours.
Introduces the fundamental ideas of computing and programming principles. Discusses a systematic approach to word problems, including analytic reading, synthesis, goal setting, planning, plan execution, and testing. Presents several models of computing, beginning with functional program design. The latter part of the course consists of two parts: a task organization (ranging from the description of data to the creation of a test suite) and a data-oriented approach to the organization of programs (ranging from atomic data to self-referential data definitions and functions as data). Offers students an opportunity to practice pair programming and public code review techniques, as found in industry today. No prior programming experience is assumed; therefore, suitable for students with little or no computer science background.

CS 5002. Discrete and Data Structures. 4 Hours.
Introduces the mathematical structures and methods that form the foundation of computer science. Studies structures such as sets, tuples, sequences, lists, trees, and graphs. Discusses functions, relations, ordering, and equivalence relations. Examines inductive and recursive definitions of structures and functions. Covers principles of proof such as truth tables, inductive proofs, and basic logic and the counting techniques and arguments needed to estimate the size of sets, the growth of functions, and the space-time complexity of algorithms. Also, discusses data structures such as arrays, stacks, queues, lists, and the algorithms that manipulate them.

CS 5003. Recitation for CS 5001. 0 Hours.
Provides a small-group discussion format to cover material in CS 5001. Coreq CS 5001.

CS 5004. Object-Oriented Design. 4 Hours.
Provides a small-group discussion format to cover material in CS 5001. Coreq CS 5001.

CS 5005. Recitation for CS 5004. 0 Hours.
Provides small-group discussion format to cover material in CS 5004.

CS 5006. Algorithms. 2 Hours.
Introduces the basic principles and techniques for the design and implementation of efficient algorithms and data representations. Considers divide-and-conquer algorithms, graph traversal algorithms, linear programming, and optimization techniques. Covers the fundamental structures for representing data, such as hash tables, trees, and graphs.

CS 5007. Computer Systems. 2 Hours.
Introduces the basic design of computing systems, computer operating systems, and assembly language using a RISC architecture. Describes caches and virtual memory. Covers the interface between assembly language and high-level languages, including call frames and pointers; the use of system calls and systems programming to show the interaction with the operating system; and the basic structures of an operating system, including application interfaces, processes, threads, synchronization, interprocess communication, deadlock, memory management, file systems, and input/output control.
CS 5010. Programming Design Paradigm. 4 Hours.
Introduces modern program design paradigms. Starts with functional program design, introducing the notion of a design recipe. The latter consists of two parts: a task organization (ranging from the description of data to the creation of a test suite) and a data-oriented approach to the organization of programs (ranging from atomic data to self-referential data definitions and functions as data). The course then progresses to object-oriented design, explaining how it generalizes and contrasts with functional design. In addition to studying program design, students also have an opportunity to practice pair-programming and public code review techniques, as found in industry today.

CS 5011. Recitation for CS 5010. 0 Hours.
Provides small-group discussion format to cover material in CS 5010.

CS 5100. Foundations of Artificial Intelligence. 4 Hours.
Introduces the fundamental problems, theories, and algorithms of the artificial intelligence field. Topics include heuristic search and game trees, knowledge representation using predicate calculus, automated deduction and its applications, problem solving and planning, and introduction to machine learning. Required course work includes the creation of working programs that solve problems, reason logically, and/or improve their own performance using techniques presented in the course. Requires experience in Java programming.

CS 5150. Game Artificial Intelligence. 4 Hours.
Offers an overview of classical and modern approaches to artificial intelligence in digital games. Focuses on the creation of believable agents and environments with the goal of providing a fun and engaging experience to a player. Covers player modeling, procedural content generation, behavior trees, interactive narrative, decision-making systems, cognitive modeling, and path planning. Explores different approaches for behavior generation, including learning and rule-based systems. Requires students to complete several individual assignments in these areas to apply the concepts covered in class. Students choose a group final project, which requires a report, to explore one aspect of artificial intelligence for games in further depth. Offers students an opportunity to learn team management and communication. Requires knowledge of algorithms and experience with object-oriented design or functional programming.

CS 5200. Database Management Systems. 4 Hours.
Introduces relational database management systems as a class of software systems. Prepares students to be sophisticated end users of database management systems. Covers design theory, query language, and performance issues. Topics include relational algebra, SQL, stored procedures, user-defined functions, cursors, embedded SQL programs, client-server interfaces, entity-relationship diagrams, normalization, B-trees, concurrency, transactions, database security, constraints, object-relational DBMSs, and specialized engines such as spatial, text, and XML conversion, and time series. Includes exercises using a commercial relational or object-relational database management system.

CS 5310. Computer Graphics. 4 Hours.
Introduces the fundamentals of two-dimensional and three-dimensional computer graphics, with an emphasis on approaches for obtaining realistic images. Covers two-dimensional algorithms for drawing lines and curves, anti-aliasing, filling, and clipping. Studies rendering of three-dimensional scenes composed of spheres, polygons, quadric surfaces, and bi-cubic surfaces using ray-tracing and radiosity. Includes techniques for adding texture to surfaces using texture and bump maps, noise, and turbulence. Requires knowledge of linear algebra.

CS 5320. Digital Image Processing. 4 Hours.
Studies the fundamental concepts of digital image processing including digitization and display of images, manipulation of images to enhance or restore image detail, encoding (compression) of images, detection of edges and other object features in images, and the formation of computed tomography (CT) images. Introduces mathematical tools such as linear systems theory and Fourier analysis and uses them to motivate and explain these image processing techniques. Requires knowledge of linear algebra.

CS 5330. Pattern Recognition and Computer Vision. 4 Hours.
Introduces fundamental techniques for low-level and high-level computer vision. Examines image formation, early processing, boundary detection, image segmentation, texture analysis, shape from shading, photometric stereo, motion analysis via optic flow, object modeling, shape description, and object recognition (classification). Discusses models of human vision (gestalt effects, texture perception, subjective contours, visual illusions, apparent motion, mental rotations, and cyclopean vision). Requires knowledge of linear algebra.

CS 5335. Robotic Science and Systems. 4 Hours.
Introduces autonomous mobile robots with a focus on algorithms and software development, including closed-loop control, robot software architecture, wheeled locomotion and navigation, tactile and basic visual sensing, obstacle detection and avoidance, and grasping and manipulation of objects. Offers students an opportunity to progressively construct mobile robots from a predesigned electromechanical kit. The robots are controlled wirelessly by software of the students’ own design, built within a provided robotics software framework. Culminates in a project that connects the algorithms and hardware developed in the course with a selected topic in the current robotics research literature.

CS 5336. Lab for CS 5335. 0 Hours.
Offers a lab section to accompany CS 5335.

CS 5340. Computer/Human Interaction. 4 Hours.
Covers the principles of human-computer interaction and the design and evaluation of user interfaces. Topics include an overview of human information processing subsystems (perception, memory, attention, and problem solving); how the properties of these systems affect the design of user interfaces; the principles, guidelines, and specification languages for designing good user interfaces, with emphasis on tool kits and libraries of standard graphical user interface objects; and a variety of interface evaluation methodologies that can be used to measure the usability of software. Other topics may include World Wide Web design principles and tools, computer-supported cooperative work, multimodal and “next generation” interfaces, speech and natural language interfaces, and virtual reality interfaces. Course work includes both the creation and implementation of original user interface designs, and the evaluation of user interfaces created by others. Requires knowledge of C programming language/UNIX.

CS 5350. Applied Geometric Representation and Computation. 4 Hours.
Surveys practical techniques for representing geometric objects in two and three dimensions, for computing their motions and interactions, and for human interfaces to manipulate them. These techniques are useful not only in graphics but also in robotics, computer vision, game design, geographic information systems, computer-aided design and manufacturing, spatial reasoning and planning, physical simulation, biomechanics, and the implementation of many types of human-computer interface. Requires undergraduate background in algorithms.

CS 5400. Principles of Programming Language. 4 Hours.
Studies the basic components of programming languages, specification of syntax and semantics, and description and implementation of programming language features. Discusses examples from a variety of languages.
CS 5500. Managing Software Development. 4 Hours.
Covers software life cycle models (waterfall, spiral, and so forth), domain engineering methods, requirements analysis methods (including formal specifications), software design principles and methods, verification and testing methods, resource and schedule estimation for individual software engineers, component-based software development methods and architecture, and languages for describing software processes. Includes a project where some of the software engineering methods (from domain modeling to testing) are applied in an example. Requires admission to MS program or completion of all transition courses.

CS 5520. Mobile Application Development. 4 Hours.
Focuses on mobile application development on a mobile phone or related platform. Discusses memory management; user interface building, including both MVC principles and specific tools; touch events; data handling, including core data, SQL, XML, and JSON; network techniques and URL loading; and, finally, specifics such as GPS and motion sensing that may be dependent on the particular mobile platform. Students are expected to work on a project that produces a professional-quality mobile application and to demonstrate the application that they have developed. The instructor chooses a modern mobile platform to be used in the course.

CS 5600. Computer Systems. 4 Hours.
Studies the structure, components, design, implementation, and internal operation of computer systems, focusing mainly on the operating system level. Reviews computer hardware and architecture including the arithmetic and logic unit, and the control unit. Covers current operating system components and construction techniques including the memory and memory controller, I/O device management, device drivers, memory management, file system structures, and the user interface. Introduces distributed operating systems. Discusses issues arising from concurrency and distribution, such as scheduling of concurrent processes, interprocess communication and synchronization, resource sharing and allocation, and deadlock management and resolution. Includes examples from real operating systems. Exposes students to the system concepts through programming exercises. Requires admission to MS program or completion of all transition courses.

CS 5610. Web Development. 4 Hours.
Discusses Web development for sites that are dynamic, data driven, and interactive. Focuses on the software development issues of integrating multiple languages, assorted data technologies, and Web interaction. Considers ASP.NET, C#, HTTP, HTML, CSS, XML, XSLT, JavaScript, AJAX, RSS/Atom, SQL, and Web services. Each student must deploy individually designed Web experiments that illustrate the Web technologies and at least one major integrative Web site project. Students may work in teams with the permission of the instructor. Each student or team must also create extensive documentation of their goals, plans, design decisions, accomplishments, and user guidelines. All source files must be open and be automatically served by a sources server.

CS 5620. Computer Architecture. 4 Hours.
Studies the design of digital computer system components including the CPU, the memory subsystem, and interconnection busses and networks. Explores modern design techniques for increasing computer system capacity. Emphasizes the growing gap between CPU and RAM speed, and the parallel operation of the growing number of functional units in a CPU. Topics include pipelining, cache, new CPU architecture models, memory bandwidth and latency, multiprocessing and parallel processing architectures, cache coherence, and memory consistency.

CS 5650. High Performance Computing. 4 Hours.
Introduces students to research in the domain of high performance computing. Each instance of this course covers a single topic with broad open questions. The required systems background needed to investigate these questions is covered in the first part of the course. Then, working in teams, students have an opportunity to address different aspects of the open questions so that in combination the entire class may learn more than any single team could accomplish. Example topics include use of new hardware such as GPUs on video boards; use of new software tools for multicore computing; development of check-pointing packages for more robust long computations; software for GUI window systems; and cloud computing.

CS 5700. Fundamentals of Computer Networking. 4 Hours.
Studies network protocols, focusing on modeling and analysis, and architectures. Introduces modeling concepts, emphasizing queuing theory, including Little’s theorem, M/M/1, M/M/m, M/D/1, and M/G/1 queuing systems. Discusses performance evaluation of computer networks including performance metrics, evaluation tools and methodology, simulation techniques, and limitations. Presents the different harmonizing functions needed for communication and efficient operation of computer networks and discusses examples of Ethernet, FDDI, and wireless networks. Covers link layer protocols including HDLC, PPP, and SLIP; packet framing; spanning tree and learning bridges, error detection techniques, and automatic repeat request algorithms; sliding window and reliable/ordered services; and queuing disciplines including FQ and WFQ. Introduces flow control schemes, such as window flow control and leaky bucket rate control schemes, and discusses congestion control and fairness. Requires knowledge of probability theory.

CS 5750. Social Computing. 4 Hours.
Offers a detailed look at popular social information systems. Studies models (both computational and sociological) of social information systems and the application of them both in theory and by analyzing real data from social network interactions. The recent popularity of online social media underlies a shift in the way people connect, communicate, and share content. When designing social computing systems, one must now understand and carefully consider the structure and use of the underlying social network. Considers questions such as: How does information spread through a social network? What mechanisms work best at encouraging collaboration?.

CS 5770. Software Vulnerabilities and Security. 4 Hours.
Seeks to help students to become aware of systems security issues and to gain a basic understanding of security. Presents the principal software and applications used in the Internet, discussing in detail the related vulnerabilities and how they are exploited. Also discusses programming vulnerabilities and how they are exploited. Examines protection and detection techniques. Includes a number of practical lab assignments as well as a discussion of current research in the field.

CS 5800. Algorithms. 4 Hours.
Presents the mathematical techniques used for the design and analysis of computer algorithms. Focuses on algorithmic design paradigms and techniques for analyzing the correctness, time, and space complexity of algorithms. Topics may include asymptotic notation, recurrences, loop invariants, Hoare triples, sorting and searching, advanced data structures, lower bounds, hashing, greedy algorithms, dynamic programming, graph algorithms, and NP-completeness.
CS 5850. Building Game Engines. 4 Hours.
Discusses the components of game engines and strategies for their software implementation. Includes graphics management algorithms (animation, scene graph, level of detail); basic artificial intelligence algorithms (search, decision making, sensing); and related algorithmic issues (networking, threading, input processing). Explores the use of data-driven software design. Offers students an opportunity to use a rendering engine and to build and integrate several software components to create a complete game engine. Requires students to work on individual assignments and then develop a project in a team, which requires a report. Offers students an opportunity to learn team/project management; work division; team communication; and the software development cycle of implementation, testing, critique, and further iteration. Requires knowledge of computer graphics, differential calculus, operating systems concepts, and algorithms.

CS 5976. Directed Study. 2-4 Hours.
Focuses on student examining standard computer science material in fresh ways or new computer science material that is not covered in formal courses. May be repeated up to three times.

CS 5978. Independent Study. 2-4 Hours.
Offers independent work under the direction of members of the department on a chosen topic. Course content depends on instructor. May be repeated up to three times.

CS 5984. Research. 2-4 Hours.
Offers an opportunity to conduct research under faculty supervision. May be repeated up to three times.