At the Khoury College of Computer Sciences, we are inspired by an increasingly interconnected society, informed by a rapidly changing job market, and focused on addressing the challenges of a complex world. Our goal is to equip students with knowledge as diverse as it is deep. Our programs provide a strong technical foundation and an essential understanding of computing concepts while integrating computer and data sciences across disciplines and industries.

Our master’s degrees are advanced programs that are designed to prepare students to be job ready through a rigorous curriculum, innovative research, experiential learning, and a collaborative environment rich in faculty expertise.

Our research-driven doctoral programs offer students an opportunity to engage in exciting projects, a vibrant community, and a challenging curriculum that offers breadth and depth in areas both within computer science and across disciplines throughout Northeastern.

Graduate education in computer science also features the top-ranked Northeastern co-op program, enabling students to supplement their classroom education with real-world experience in the field. The college has over 450 industry partners, including:

- Amazon
- Bloomberg
- Chewy
- Facebook
- Fidelity Investments
- Goldman Sachs
- Hubspot
- Intuit
- Microsoft
- MITRE
- Red Hat
- Rue La La
- Toast
- Trip Advisor
- Verizon Communications

**Programs**

**Doctor of Philosophy (PhD)**

- Computer Science (http://catalog.northeastern.edu/graduate/ computer-information-science/computer-science/computer-science- phd/)
- Computer Science—Advanced Entry (http://catalog.northeastern.edu/graduate/computer-information-science/ computer-science/computer-science-phd-advanced/)

**Master of Science (MS)**

- Artificial Intelligence (http://catalog.northeastern.edu/graduate/ computer-information-science/computer-science/artificial- intelligence-ms/)
- Data Science (http://catalog.northeastern.edu/graduate/computer- information-science/computer-science/data-science-ms/)
- Health Data Analytics (http://catalog.northeastern.edu/graduate/ computer-information-science/computer-science/health-data- analytics-ms/)
- Robotics (http://catalog.northeastern.edu/graduate/engineering/ mechanical-industrial/robotics-ms/)

**Master of Science in Computer Science (MSCS)**

- Computer Science (http://catalog.northeastern.edu/graduate/ computer-information-science/computer-science/computer-science- mscs/)
- Computer Science—Align Program (http://catalog.northeastern.edu/ graduate/computer-information-science/computer-science/ computer-science-align-program-mscs/)

**Graduate Certificate**

- Computer Science (http://catalog.northeastern.edu/graduate/ computer-information-science/computer-science/computer-science- graduate-certificate/)
- Data Analytics (http://catalog.northeastern.edu/graduate/computer- information-science/computer-science/data-analytics-graduate- certificate/)

**Courses**

**Computer Science Courses**

Search CS, DS, IS Courses using FocusSearch (http://catalog.northeastern.edu/class-search/?subject=CS%2CDS%2CIS/)

**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 proof, 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.

Presents a comparative approach to object-oriented programming and design. Discusses the concepts of object, class, metaclass, message, method, inheritance, and genericity. Reviews forms of polymorphism in object-oriented languages. Contrast the use of inheritance and composition as dual techniques for software reuse—forwarding vs. delegation and subclassing vs. subtyping. Offers students an opportunity to obtain 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). Illustrates basic concepts in object-oriented design with case studies in application frameworks and by writing programs in Java.

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 5081. Accessibility and Inclusiveness in Human-Computer Interaction. 2 Hours.

Covers the fundamentals of user experience. Emphasizes how to integrate accessibility and usability to enhance the user experience in everyday products and services for everyone, including seniors and people with disabilities.

CS 5082. Privacy and Security of User Accounts: Patterns and Best Practices. 2 Hours.

Introduces approaches for authentication (ensuring you know who someone is) and authorization (ensuring they have access to a given resource or service). Studies how to identify relevant issues from the consumer or user side of account creation and management; identify expectations and liabilities for the developer or company providing a user-based account; share existing software design patterns and technologies to help you implement secure user accounts, including OAuth and anonymous accounts; and discusses UX design issues around user account creation and maintenance. Relevant for anyone who wants to create an application or service with a user registration and login page. Covers why you don’t want to build this functionality yourself and how you can use existing tools and technologies that shield you from liability for storing user data.

CS 5083. Software Project Management with Scrum. 2 Hours.

Offers students an opportunity to obtain an understanding of the Scrum methodology for managing software projects using lean principles. Explains the Scrum framework as well as key ceremonies and roles. Shows which aspects of Scrum are required and how they manage project risk.

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 5180. Reinforcement Learning and Sequential Decision Making. 4 Hours.

Introduces reinforcement learning and the underlying computational frameworks and the Markov decision process framework. Covers a variety of reinforcement learning algorithms, including model-based, model-free, value function, policy gradient, actor-critic, and Monte Carlo methods. Examines commonly used representations including deep learning representations and approaches to partially observable problems. Students are expected to have a working knowledge of probability and linear algebra, to complete programming assignments, and to complete a course project that applies some form of reinforcement learning to a problem of interest.
CS 5200. Database Management Systems. 4 Hours.
Introduces relational database management systems as a class of software systems. Prepares students to be sophisticated users of database management systems. Covers design theory, query language, and performance/tuning 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, 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 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 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. Foundations of Software Engineering. 4 Hours.
Covers the foundations of software engineering, including software development life cycle models (e.g., waterfall, spiral, agile); requirements analysis; user-centered design; software design principles and patterns; testing (functional testing, structural testing, testing strategies); code refactoring and debugging; software architecture and design; and integration and deployment. Includes a course project where some of the software engineering methods (from requirements analysis to testing) are applied in a team-based setting. 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 source server.

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 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 5964. Experiential Project. 0 Hours.
Offers students an applied project setting in which to apply their curricular learning. Working with a sponsor, students refine an applied research topic, perform research, develop recommendations that are shared with a partner sponsor, and create a plan for implementing their recommendations. Seeks to benefit students with a curriculum that supports the development of key business communication skills, project and client management skills, and frameworks for business analysis. Offers students an opportunity to learn from sponsor feedback, review ‘lessons learned’; and incorporate suggestions from this review to improve and further develop their career development and professional plan. May be repeated twice.

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 6110. Knowledge-Based Systems. 4 Hours.
Focuses on the acquisition, organization, and use of world knowledge in computers, and the challenge of creating programs with common sense. Topics include knowledge representation and reasoning models beyond predicate calculus, Bayesian inference and other models of reasoning and decision making under uncertainty, rule-based expert systems, case-based and analogical reasoning, and introduction to natural language processing. Course work includes the creation of working programs that store and manipulate world knowledge using techniques presented in the course.

CS 6120. Natural Language Processing. 4 Hours.
Provides an introduction to 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 corporuses, 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.

CS 6130. Affective Computing. 4 Hours.
Studies affective computing—computing that relates to, arises from, or influences emotions. Offers an overview of the theory of human emotion (how it arises from and influences cognition, the body, and the social environment) and computational techniques for modeling human emotion processes as well as for recognizing and synthesizing emotional behavior. Discusses how these can be applied to application design. Offers students an opportunity to gain a strong background in the theory and practice of human-centered computing as it relates to games, immersive environments, and pedagogical applications. Brings together students from different disciplines to work together and learn from each other. CS 6130 and PSYC 6130 are cross-listed.

CS 6140. Machine Learning. 4 Hours.
Provides a broad look at a variety of techniques used in machine learning and data mining, and also examines issues associated with their use. Topics include algorithms for supervised learning including decision tree induction, artificial neural networks, instance-based learning, probabilistic methods, and support vector machines; unsupervised learning; and reinforcement learning. Also covers computational learning theory and other methods for analyzing and measuring the performance of learning algorithms. Course work includes a programming term project.
CS 6200. Information Retrieval. 4 Hours.
Provides an introduction to information retrieval systems and different approaches to information retrieval. Topics covered include evaluation of information retrieval systems; retrieval, language, and indexing models; file organization; compression; relevance feedback; clustering; distributed retrieval and metasearch; probabilistic approaches to information retrieval; Web retrieval; filtering, collaborative filtering, and recommendation systems; cross-language IR; multimedia IR; and machine learning for information retrieval.

CS 6220. Data Mining Techniques. 4 Hours.
Covers various aspects of data mining, including classification, prediction, ensemble methods, association rules, sequence mining, and cluster analysis. The class project involves hands-on practice of mining useful knowledge from a large data set.

CS 6240. Large-Scale Parallel Data Processing. 4 Hours.
Covers big-data analysis techniques that scale out with increasing number of compute nodes, e.g., for cloud computing. Emphasizes approaches for problem and data partitioning that distribute work effectively, while keeping total cost for computation and data transfer low. Studies and analyzes deterministic and random algorithms from a variety of domains, including graphs, data mining, linear algebra, and information retrieval in terms of their cost, scalability, and robustness against skew. Course work emphasizes hands-on programming experience with modern state-of-the-art big-data processing technology. Students who do not meet course prerequisites may seek permission of instructor.

CS 6250. Empirical Research Methods. 4 Hours.
Presents an overview of methods for conducting empirical research within computer science. These methods help provide objective answers to questions about the usability, effectiveness, and acceptability of systems. The course covers the basics of the scientific method, building from a survey of objective measures to the fundamentals of hypothesis testing using relatively simple research designs, and on to more advanced research designs and statistical methods. The course also includes a significant amount of fieldwork, spanning the design, conduct, and presentation of small empirical studies.

CS 6350. Lab for CS 6350. 0 Hours.
Accompanies CS 6350. Covers topics from the course through various experiments.

CS 6410. Compilers. 4 Hours.
Expect each student to write a small compiler. Topics include parser generation, abstract syntax trees, symbol tables, type checking, generation of intermediate code, simple code improvement, register allocation, run-time structures, and code generation.

CS 6510. Advanced Software Development. 4 Hours.
Designed to integrate academic concepts and practical experience of software design by having students work as part of a programming team, with an option to lead a subteam. Offers students an opportunity to study, in-depth, some aspects of the development process. The goal is to have students participate in a large-scale project, taking time to reflect and analyze the work and the process, rather than concentrating exclusively on the final product. Students who do not meet course prerequisites may seek permission of instructor.

CS 6535. Engineering Reliable Software. 4 Hours.
Continues the exploration of several themes from CS 5010: unit testing, random testing, and logical reasoning about software. Specifically revisits the idea of systematic design and its connection to making logical claims about the workings of programs. After an introduction to the ACL2 programming language and theorem prover, offers students an opportunity to redesign interactive games (e.g., “Space Invaders”) and work on turning them into reliable projects. Students who do not meet course prerequisites may seek permission of instructor.

CS 6620. Fundamentals of Cloud Computing. 4 Hours.
Covers fundamentals of cloud computing, including virtualization and containers, distributed file systems and object stores, infrastructure as a service platforms, open source cloud platforms, key big data platforms, and topics in data center scale systems. Combines classroom material delivered via lectures, readings from literature, student presentations, and a semester-long software project.

CS 6650. Building Scalable Distributed Systems. 4 Hours.
Covers the essential elements of distributed, concurrent systems and builds upon that knowledge with engineering principles and practical experience with state-of-the-art technologies and methods for building scalable systems. Scalability is an essential quality of internet-facing systems and requires specialized skills and knowledge to build systems that scale at low cost.

CS 6710. Wireless Network. 4 Hours.
Covers both theoretical issues related to wireless networking and practical systems for both wireless data networks and cellular wireless telecommunication systems. Topics include fundamentals of radio communications, channel multiple access schemes, wireless local area networks, routing in multihop ad hoc wireless networks, mobile IP, and TCP improvements for wireless links, cellular telecommunication systems, and quality of service in the context of wireless networks. Requires a project that addresses some recent research issues in wireless and mobile networking.

CS 6760. Privacy, Security, and Usability. 4 Hours.
Challenges conventional wisdom and encourages students to discover ways that security, privacy, and usability can be made synergistic in system design. Usability and security are widely seen as two antagonistic design goals for complex computer systems. Topics include computer forensics, network forensics, user interface design, backups, logging, economic factors affecting adoption of security technology, trust management, and related public policy. Uses case studies such as PGP, S/MIME, and SSL. Introduces basic cryptography and hash function as it is needed. Course work includes analysis of papers, problem sets, and a substantial term project.

CS 6800. Application of Information Theory. 4 Hours.
Introduces information theory and its applications to various computational disciplines. Covers the basic concepts of information theory, including entropy, relative entropy, mutual information, and the asymptotic equipartition property. Concentrates on applications of information theory to computer science and other computational disciplines, including compression, coding, Markov chains, machine learning, information retrieval, statistics, computational linguistics, computational biology, wired and wireless networks, and image and speech processing. The course is self-contained; no prior knowledge of information theory is required or assumed. Requires an undergraduate course in probability.
CS 6949. Graduate Cooperative Education Seminar. 1 Hour.
Intended to prepare graduate students in computer and information science for co-op. Topics include résumé writing, interviewing, job search strategy, ethics, professional behavior, and the college’s co-op policies. Students intending to participate in a co-op or internship must satisfactorily complete this course, which is typically taken during the student’s first semester.

CS 6954. Co-op Work Experience - Half-Time. 0 Hours.
Provides eligible students with an opportunity for work experience. May be repeated without limit.

CS 6962. Elective. 1–4 Hours.
Offers elective credit for courses taken at other academic institutions. May be repeated without limit.

CS 6964. Co-op Work Experience. 0 Hours.
Provides eligible students with an opportunity for work experience. May be repeated without limit.

CS 6965. Co-op Work Experience Abroad. 0 Hours.
Provides eligible students with an opportunity for work experience abroad. May be repeated without limit.

CS 7140. Advanced Machine Learning. 4 Hours.
Covers topics in advanced machine learning. Presents materials in the current machine learning literature. Focuses on graphical models, latent variable models, Bayesian inference, and nonparametric Bayesian methods. Seeks to prepare students to do research in machine learning. Extends students to read conference and journal articles, present these articles, and write an individual research paper. CS 7140 and ECE 7397 are cross-listed.

CS 7170. Seminar in Artificial Intelligence. 2–4 Hours.
Gives students the opportunity to read and present various survey and research papers in artificial intelligence. May be repeated for credit for PhD students; faculty supervisor and topics vary from semester to semester.

CS 7180. Special Topics in Artificial Intelligence. 4 Hours.
Offers various topics on artificial intelligence. May be repeated up to two times.

CS 7240. Principles of Scalable Data Management: Theory, Algorithms, and Database Systems. 4 Hours.
Covers the algorithms, core principles, and foundational concepts for managing data at scale. Topics include data models, query languages, query execution and optimization, complexity of query execution and query resilience, data stream processing, parallel data processing, transactions, linear vs. relational algebra, factorization, and uncertainty in logic. Requires standard CS knowledge of algorithms and hardness (e.g., a typical undergraduate class based on a standard algorithms textbook such as Ericson; Cormen, Leiserson, Rivest, and Stein; or Dasgupta, Papadimitriou, and Vazirani). Offers students an opportunity to gain hands-on experience through smaller assignments and a project. The project is flexible to allow students to explore scalable data management and analysis aspects related to their PhD research.

CS 7250. Information Visualization: Theory and Applications. 4 Hours.
Covers foundational as well as contemporary topics of interest in data visualization to enable the effective representation of data across disciplines, including examples drawn from computer science, physical sciences, biomedical sciences, humanities, and economics. Topics include data visualization theory and methodology, visualization design and evaluation, visual perception and cognition, interaction principles, and data encoding and representation techniques. Students who do not meet course restrictions may seek permission of instructor.

CS 7260. Visualization for Network Science. 4 Hours.
Covers the principles of information visualization in the specific context of network science. Introduces visual encoding of data and our understanding of human vision and perception; interaction principles including filtering, pivoting, aggregation; and both quantitative and human subjects evaluation techniques. Covers visualization techniques for several network types, including multivariate networks with attributes for entities and relationships, evolving and dynamic networks that change over time, heterogeneous networks with multiple types of entities, and geospatial networks. Offers students an opportunity to learn about the design of layout algorithms for node-link and matrix visualizations.

CS 7290. Special Topics in Data Science. 4 Hours.
Offers special topics in data science, including machine learning, statistics, data mining, parallel and distributed data analysis, database systems, information retrieval, knowledge representation, information visualization, natural language processing, computational biology and bioinformatics, computational social science, digital humanities, health informatics, business, and predictive analytics. May be repeated once for up to 8 total credits.

CS 7340. Theory and Methods in Human Computer Interaction. 4 Hours.
Covers the foundations of human abilities, computational artifacts, design, and evaluation. Human computer interaction concerns the design and evaluation of software based on a deep understanding of how humans interact with computers, devices, and sensors. The field blends theories from psychology and computer science, using methods from AI and design. Introduces cognitive, perceptual, and affective theories and theories of individual differences that allow us to design and develop better computer software and systems. Also covers research methods for designing and evaluating computer software systems. Topics discussed in the context of next-generation interaction modalities include sensors, haptics, wearables, and performative interfaces. Students who do not meet course restrictions may seek permission of instructor.

CS 7390. Special Topics in Human-Centered Computing. 4 Hours.
Offers various topics in human-centered computing.

CS 7400. Intensive Principles of Programming Languages. 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 7480. Special Topics in Programming Language. 4 Hours.
Offers various topics in programming language. May be repeated up to two times.

CS 7485. Special Topics in Formal Methods. 4 Hours.
Offers various topics in formal methods. May be repeated without limit.

CS 7580. Special Topics in Software Engineering. 4 Hours.
Offers various topics on software engineering. May be repeated up to two times.
CS 7600. Intensive Computer Systems. 4 Hours.
Studies the structure, components, design, implementation, and internal operation of computer systems, focusing 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. Discusses distributed operating systems, real-time systems, and addresses concurrent processes, scheduling, interprocess communication, and synchronization. Discusses relevant distributed algorithms. Also covers design and analysis techniques for desirable properties in computer systems including functional correctness (in the absence of faults), performance and throughput, fault-tolerance and reliability, real-time response, security, and quality of service. Draws examples from real operating systems. Emphasizes abstraction, while programming exercises are used to facilitate the understanding of concepts.

CS 7610. Foundations of Distributed Systems. 4 Hours.
Covers foundational concepts in the design and implementation of efficient and reliable distributed computing systems. Covers internet communication protocols, fault-tolerant computing, synchronization protocols, synchronous and asynchronous computing, dynamic group communication systems, load balancing, Byzantine models, distributed hash tables, distributed file systems, and application of foundational concepts to modern distributed systems in the field. Requires knowledge of operating systems; e.g., an undergraduate course in Systems and Networks, Computer Systems, or Networks and Distributed systems.

CS 7680. Special Topics in Computer Systems. 4 Hours.
Offers various topics on computer systems. May be repeated up to two times.

CS 7775. Seminar in Computer Security. 2-4 Hours.
Gives students the opportunity to read and present various survey and research papers in cryptography and computer security. Faculty supervisor and topics vary from semester to semester. May be repeated for credit for PhD students.

CS 7780. Special Topics in Networks. 4 Hours.
Offers various topics on networks. May be repeated up to two times.

CS 7800. Advanced Algorithms. 4 Hours.
Presents advanced mathematical techniques for designing and analyzing computer algorithms. Reviews some of the material covered in CS 5800 and then covers advanced topics. Emphasizes theoretical underpinnings of techniques used to solve problems arising in diverse domains. Topics include asymptotic analysis, advanced data structures, dynamic programming, greedy algorithms and matroid theory, amortized analysis, randomization, string matching, algebraic algorithms, and approximation algorithms. Introduces Turing machines, P and NP classes, polynomial-time reducibility, and NP completeness.

CS 7805. Theory of Computation. 4 Hours.
Examines formal models of computation, notions of undecidability, and basic complexity theory. Models of computation include finite state automata, pushdown automata, and Turing machines. Discusses the properties of regular sets and context-free languages. Also covers partial recursive functions, primitive recursive functions, recursively enumerable sets, Turing decidability, and unsolvable problems. Discusses the concept of reductions, time and space complexity classes, and the polynomial-time hierarchy.

CS 7810. Foundations of Cryptography. 4 Hours.
Offers students at the PhD level an accelerated introduction to cryptography and quickly progresses to advanced topics that are at the forefront of current research. Cryptography is the science of protecting information against adversarial eavesdropping and tampering. Examines what kind of security properties can be achieved by relying solely on probability and information theory, without restricting the adversary’s computational power. Studies the complexity-theoretic basis of modern cryptography and the connection between computational hardness and pseudo-randomness. Explores, as the main component of the course, how to take a few well-studied problems in number theory and algebra and use them to build powerful cryptosystems with advanced functionality and security properties. Requires prior completion of an undergraduate course in the theory of computation (Northeastern’s CS 3800 or equivalent).

CS 7880. Special Topics in Theoretical Computer Science. 4 Hours.
Covers various topics including advanced cryptography, approximation algorithms, complexity theory, computational algebra, distributed computing, formal verification, network algorithms, online computation, parallel computing, and randomness and computation. May be repeated up to two times.

CS 7962. Elective. 2-4 Hours.
Offers elective credit for courses taken at other academic institutions. May be repeated without limit.

CS 7976. 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 without limit.

CS 7990. Thesis. 4 Hours.
Offers selected work with the agreement of a project supervisor. May be repeated without limit.

CS 7996. Thesis Continuation. 0 Hours.
Offers continued thesis work conducted under the supervision of a departmental faculty.

CS 8674. Master’s Project. 4 Hours.
Offers selected work with the agreement of a project supervisor. May be repeated once.

CS 8949. Research Work Experience. 0 Hours.
Provides an opportunity for all doctoral students to engage in industry research in the area of their dissertation. Doctoral students register for this course before starting their off-campus internships. May be repeated without limit.

CS 8982. Readings. 1-8 Hours.
Offers selected readings under the supervision of a faculty member. May be repeated without limit.

CS 9000. PhD Candidacy Achieved. 0 Hours.
Indicates successful completion of the doctoral comprehensive exam.

CS 9990. Dissertation. 0 Hours.
Offers selected work with the agreement of a thesis supervisor. May be repeated once.

CS 9996. Dissertation Continuation. 0 Hours.
Continues work with the agreement of a thesis supervisor. May be repeated without limit.
DS 5010. Introduction to Programming for Data Science. 4 Hours.
Offers an introductory course on fundamentals of programming and data structures. Covers lists, arrays, trees, hash tables, etc.; program design, programming practices, testing, debugging, maintainability, data collection techniques, and data cleaning and preprocessing. Includes a class project, where students use the concepts covered to collect data from the web, clean and preprocess the data, and make it ready for analysis.

DS 5020. Introduction to Linear Algebra and Probability for Data Science. 4 Hours.
Offers an introductory course on the basics of statistics, probability, and linear algebra. Covers random variables, frequency distributions, measures of central tendency, measures of dispersion, moments of a distribution, discrete and continuous probability distributions, chain rule, Bayes’ rule, correlation theory, basic sampling, matrix operations, trace of a matrix, norms, linear independence and ranks, inverse of a matrix, orthogonal matrices, range and null-space of a matrix, the determinant of a matrix, positive semidefinite matrices, eigenvalues, and eigenvectors.

DS 5110. Introduction to Data Management and Processing. 4 Hours.
Discusses the practical issues and techniques for data importing, tidying, transforming, and modeling. Offers a gentle introduction to techniques for processing big data. Programming is a cross-cutting aspect of the course. Offers students an opportunity to gain experience with data science tools through short assignments. Course work includes a term project based on real-world data. Covers data management and processing—definition and background; data transformation; data import; data cleaning; data modeling; relational and analytic databases; basics of SQL; programming in R and/or Python; MapReduce fundamentals and distributed data management; data processing pipelines, connecting multiple data management and analysis components; interaction between the capabilities and requirements of data analysis methods (data structures, algorithms, memory requirements) and the choice of data storage and management tools; and repeatable and reproducible data analysis.

DS 5220. Supervised Machine Learning and Learning Theory. 4 Hours.
Introduces supervised machine learning, which is the study and design of algorithms that enable computers/machines to learn from experience or data, given examples of data with a known outcome of interest. Offers a broad view of models and algorithms for supervised decision making. Discusses the methodological foundations behind the models and the algorithms, as well as issues of practical implementation and use, and techniques for assessing the performance. Includes a term project involving programming and/or work with real-life data sets. Requires proficiency in a programming language such as Python, R, or MATLAB.

DS 5230. Unsupervised Machine Learning and Data Mining. 4 Hours.
Introduces unsupervised machine learning and data mining, which is the process of discovering and summarizing patterns from large amounts of data, without examples of data with a known outcome of interest. Offers a broad view of models and algorithms for unsupervised data exploration. Discusses the methodological foundations behind the models and the algorithms, as well as issues of practical implementation and use, and techniques for assessing the performance. Includes a term project involving programming and/or work with real-life data sets. Requires proficiency in a programming language such as Python, R, or MATLAB.

DS 5500. Information Visualization: Applications in Data Science. 4 Hours.
Offers students an opportunity to develop effective communication skills with data by drawing from different disciplines including physics, biology, health science, social science, geography, business, and economics. Introduces principles of effective oral and written communication and a wide range of visual data encodings and representations. Covers the foundational principles for visual representations, including human vision and perception and basic interactivity. A semester-long project requires students to translate the domain science or technology problem into the language of data science; design, evaluate, implement, and deploy both static and interactive visualizations of data and data analysis results; translate the results into the language of the original science or technology problem; communicate the findings in oral and written form; and provide constructive criticism of other examples of data communication and visualization.

DS 6050. Seminar in Data Science. 4 Hours.
Offers students an opportunity to learn how to approach data analysis problems in a systematic manner and to learn how to design data analysis pipelines, as well as how to implement them at scale in the context of real-world problems. Data science is at the intersection of statistics, machine learning, and software development. Data analysis problems are solved in a series of datacentric steps: data acquisition, data cleaning, data transformation, data modelling, and data visualization.

DS 7995. Project. 1-4 Hours.
Offers students an opportunity to participate in a direct data science project under the supervision of a faculty member. May be repeated once for a total of 8 credits.