

# COMPUTER SCIENCE

[mccormick.northwestern.edu/computer-science](https://mccormick.northwestern.edu/computer-science)

Computer science involves the understanding, use, and extension of computational ideas and their implementation. A Northwestern computer science graduate will

- Comprehend the breadth of computer science, its key intellectual divisions and questions, and its past and likely future influence on engineering, science, medicine, business, and law
- Approach problems from the algorithmic perspective, understanding the nature and broad reach of computation and how to apply it abstractly
- Approach problems from the systems perspective, understanding the evolving layers of the software/hardware stack and how to create, use, and extend them
- Approach problems from the perspective of artificial intelligence, understanding how to make progress in solving seemingly intractable problems
- Design and implement complex software systems, individually and as a team member
- Design and implement effective human-machine interfaces

Courses and undergraduate research opportunities focus on software, ranging from theoretical models to practical applications. They establish a common breadth of knowledge in computer science, allowing students flexibility in areas in which they choose to specialize, such as

- *Artificial intelligence*, including mobile robots with perceptual systems, models of memory and reasoning, knowledge representation, natural-language comprehension, planning, and problem solving
- *Computer systems*, including parallel, distributed, and real-time systems, performance evaluation, prediction, and scheduling
- *Networked systems*, including peer-to-peer computing, large-scale data storage, network security, and pervasive computing environments
- *Programming languages and compilers*, including semantics, optimization, and software
- *Human-computer interaction*, including interface design, task modeling, intelligent interfaces, and authoring tools
- *Distributed interactive systems*, including client-server and web-based applications such as heterogeneous databases and multimedia learning environments
- *Theoretical computer science*, focusing on algorithm design and analysis of algorithms' worst- and average-case behavior
- *Intelligent information systems*, including "frictionless" proactive systems and context- and task-sensitive retrieval systems
- *Computer graphics and human-computer interfaces* for spatial applications, visualization, and computer entertainment

## Programs of Study

- Computer Science Degree (<https://catalogs.northwestern.edu/undergraduate/engineering-applied-science/computer-science/computer-science-degree/>)
- Computer Science Minor (McCormick School of Engineering) (<https://catalogs.northwestern.edu/undergraduate/engineering-applied-science/computer-science/computer-science-minor/>)

**COMP\_SCI 101-0 Computer Science: Concepts, Philosophy, and Connections (1 Unit)** General introduction to historical and current intellectual questions in computer science. Theory, systems, artificial intelligence, interfaces, software development, and interactions with business, politics, law, medicine, engineering, and other sciences. *Social Behavioral Sciences Distro Area*

**COMP\_SCI 110-0 Introduction to Computer Programming (1 Unit)** Introduction to programming practice using a modern programming language. Analysis and formulation of problems for computer solution. Systematic design, construction, and testing of programs. Substantial programming assignments. Not to be taken for credit with or after COMP\_SCI 111-0. *Empirical and Deductive Reasoning Foundational Dis Formal Studies Distro Area*

**COMP\_SCI 111-0 Fundamentals of Computer Programming (1 Unit)** Fundamental concepts of computer programming with heavy emphasis on design of recursive algorithms and test-driven development. Functional, imperative, and object-oriented programming paradigms. Procedural abstraction, data abstraction, and modularity. Required for the computer science degree. *Empirical and Deductive Reasoning Foundational Dis Formal Studies Distro Area*

**COMP\_SCI 111-SG Computer Science 111-SG Peer-Guided Study Group (0 Unit)** Peer-guided study group for students enrolled in COMP\_SCI 111. Meets weekly in small groups, along with a peer facilitator, to collaboratively review material, work through practice problems, and clarify course concepts. Enrollment optional. Must be enrolled in COMP\_SCI 111-0 to enroll in this study group.

**COMP\_SCI 150-0 Fundamentals of Computer Programming 1.5 (1 Unit)** An introduction to Object-oriented programming: focus on Python but including a brief introduction to a statically typed language (e.g. C++). Students will use some approaches from Artificial Intelligence and Machine Learning to complete programming assignments. Required for the computer science degree. Prerequisite: COMP\_SCI 110-0 or COMP\_SCI 111-0 or GEN\_ENG 205-1 or GEN\_ENG 206-1 or CS MS. *Empirical and Deductive Reasoning Foundational Dis Formal Studies Distro Area*

**COMP\_SCI 211-0 Fundamentals of Computer Programming II (1 Unit)** CS 211 teaches foundational software design skills at a small-to-medium scale. We aim to provide a bridge from the student-oriented How to Design Programs languages to real, industry-standard languages and tools. In the first half of the course, you'll learn the basics of imperative programming and manual memory management using the C programming language. In the second half of the course, we'll transition to C++, which provides abstraction mechanisms such as classes and templates that we use to express our design ideas. Topics include expressions, statements, types, functions, branches and iteration, user-defined types, data hiding, basic UNIX shell usage, and testing. Prerequisite: COMP\_SCI 111-0 and COMP\_SCI 150-0.

**COMP\_SCI 212-0 Math Foundations of CS Part 1: Discrete Math for CS (1 Unit)** Basic concepts of finite and structural mathematics. Sets, axiomatic systems, the propositional and predicate calculi, and graph theory. Application to computer science: sequential machines, formal grammars, and software design. Syllabus - <https://www.mccormick.northwestern.edu/computer-science/academics/courses/descriptions/212.html>. Prerequisite: CS 110 or CS 111.

**COMP\_SCI 213-0 Introduction to Computer Systems (1 Unit)** The hierarchy of abstractions and implementations that make up a modern computer system; demystifying the machine and the tools used to program it; systems programming in C in the UNIX

environment. Preparation for upper-level systems courses. Prerequisite: COMP\_SCI 211-0.

**COMP\_SCI 214-0 Data Structures & Algorithms (1 Unit)** Design, implementation, and performance analysis of abstract data types; data structures and their algorithms. Topics include fundamental collection classes, tree and graph representations and walks, search trees, sorting, priority queues and heaps, least-cost paths computations, and disjoint-set structures. Required for the computer science degree. Prerequisite: COMP\_SCI 111 and (COMP\_SCI 150 or COMP\_SCI 211). Prerequisites are strict. Stat 201 is \*not\* a substitute for Comp\_Sci 111.

**COMP\_SCI 217-0 Data Management & Information Processing (1 Unit)** This class offers a hands-on introduction to data representation, data modelling, and the SQL language for accessing and analyzing data in relational databases. Students access and analyze data in real-world large-scale databases from the public domain. Not for computer science or computer engineering degree candidates. Prerequisite: COMP\_SCI 110-0 or COMP\_SCI 111-0 or COMP\_SCI 150-0 or COMP\_SCI 211-0 or consent of instructor.

**COMP\_SCI 260-0 Introduction to Law and Digital Technologies (1 Unit)** Course summary: This course explores the legal implications of the contemporary technology landscape, including the growth of artificial intelligence, the ecosystem for creating and disseminating digital information, and the challenges of ensuring digital privacy and algorithm equity. The course aims to help students develop a broad, contextualized view of the legal and policy opportunities and challenges associated with rapid technological change. A key goal of the course is for students to acquire the skills to understand, contribute to, and shape the dialog on complex issues at the intersection of technology and law. Prerequisites: N/A. *Ethical and Evaluative Thinking Foundational Disci Ethics Values Distro Area*

**COMP\_SCI 262-0 Mathematical Foundations of Computer Science - Part 2 (1 Unit)** Mathematical Foundations of Computer Science (MFCS) sequence covers mathematical topics of probability, linear algebra, multivariable calculus and basic optimization that are crucial for many areas of modern computer science, including data science and machine learning. Unlike Part I of the MFCS sequence (CS212), this course will focus more on the above areas of continuous mathematics that are useful in computer science. Prerequisite: COMP\_SCI 212-0 or MATH 300-0.

**COMP\_SCI 295-0 Special Topics in Computer Science (1 Unit)** Topics suggested by students or faculty and approved by the department.

**COMP\_SCI 296-0 Intermediate Topics in Computer Science (1 Unit)** Topics suggested by faculty and approved by the department. Intended to apply toward advanced elective for the computer science major.

**COMP\_SCI 298-0 CS Research Track Program (1 Unit)** Topics suggested by faculty and approved by the department. Equivalent to CS 396 but intended to apply toward advanced elective for the computer science major. Prerequisites: Consent of Instructor.

**COMP\_SCI 301-0 Introduction to Robotics Laboratory (1 Unit)** Lab-based introduction to robotics, focusing on hardware (sensors/actuators) and software (sensor processing/behavior development); motion control and planning; artificial intelligence; machine learning. Not open to graduate students except by consent of instructor. Prerequisites: COMP\_SCI 110 or COMP\_SCI 111 or COMP\_SCI 150 or MECH\_ENG 224, or permission of instructor.

**COMP\_SCI 302-0 Artificial Life (1 Unit)** This course introduces students to the growing field of Artificial Life: the study of "life as it could be" rather than as it happened to unfold on

Earth. We will address fundamental questions about what distinguishes life from matter; the conditions in which complex adaptive systems may arise and how they may be shaped to suit our needs; as well as whether or not there can even be a science of the artificial. Answers to these questions are of interest as a basic intellectual pursuit, but they may also lead to useful computational, mechanical, chemical and biological technologies.

Prerequisites: Comp\_sci 349 or MS CS or PhDs CS.

**COMP\_SCI 303-0 Full Stack Software Engineering (1 Unit)** Modern software frameworks such as Next.js, Django, and Spring Boot provide the foundational components to quickly develop and deploy full-stack software applications. These frameworks, which are reusable abstractions of code wrapped in well-defined Application Programming Interfaces (APIs), provide several benefits for software developers, such as easier development and maintenance, standardization and compatibility, improved security, increased productivity, and easier collaboration. In this course we will explore one of these frameworks to create, iteratively refine, and deploy a real-world application. Prerequisite: Students must have completed COMP\_SCI 213-0 and 214-0 to enroll. CS MS and PhD students must receive instructor consent.

**COMP\_SCI 307-0 Introduction to Cryptography (1 Unit)** This course covers the basic knowledge in understanding and using cryptography. The main focus is on definitions, theoretical foundations, and rigorous proofs of security, with some programming practice. Topics include symmetric and public-key encryption, message integrity, hash functions, block-cipher design and analysis, number theory, and digital signatures.

Prerequisite: COMP\_SCI 212 or Instructor Consent.

**COMP\_SCI 308-0 Foundations of Security (1 Unit)** "Cybersecurity can be an intimidating word, but it doesn't need to be. It is actually fundamental to every application or system we use, however small. One does not need to be "interested" in security to be accountable for creating secure programs; anyone who will be responsible for maintaining or building any computer systems will be accountable. Have you been or do you know anyone who has been a victim of a data breach? Data breaches have become commonplace with the causes often being a simple security mistake/piece of insecure infrastructure with a massive impact. Do you see yourself building a simple application that handles user data? You would have a responsibility to protect your users' data. Are you interested in cybersecurity as a specialization or career path? You'll probably get to advise the design of systems to be secure from the ground up and mitigate security incidents in real time. And finally: do you envision yourself in a role where you will either build or manage any systems that people will use? Then your systems need to be secure. Systems, software, and tools that we use are all vulnerable to security breaches and incidents. However, security, historically and still today, is often an afterthought in the design of computer systems, which only exacerbates their vulnerability. And while cybersecurity itself is a huge and exciting field for those interested, there are core aspects of security that are accessible to and necessary for anyone in a computing-related occupation.

Prerequisite: COMP\_SCI 211 and 214 or Instructor Consent".

**COMP\_SCI 310-0 Scalable Software Architectures (1 Unit)** Teaches software design principles for building high-scale Internet services. Focuses on challenges arising when assembling software services that run on many machines in parallel and which require the coordination of multiple software applications. Prerequisites: CS 213 and 214 or CS MS or CS PhDs or Instructor permission.

**COMP\_SCI 311-0 Inclusive Making (1 Unit)**

Inclusive Making is about centering disability within computer science. The class explores the promises and shortcomings of making through a critical disability studies lens. It also looks at existing making practices within disability communities. Throughout the class, students reflect on their assumptions about disability and computer science, and wrestle with tensions related to making and accessibility alongside community organizations. Taught with LRN\_SCI 309-0, students cannot enroll and receive credit for both.

Prerequisites: COMP\_SCI 110 or COMP\_SCI 111 or COMP\_SCI MS or COMP\_SCI PhDs or Instructor consent.

#### **COMP\_SCI 312-0 Data Privacy (1 Unit)**

Data breaches, privacy breaches, and concerns about algorithmic decision-making have been on the rise. As a result, data privacy has become an increasingly significant concern in the past several years. Individuals and organizations often trust institutions with their data with the expectation that one's data is private from others or to the handling institutions and that it is not used for unfair practices. To ensure the privacy of sensitive data, privacy mechanisms have been developed to preserve the privacy of data without reducing its functionality. The goal of this course is to introduce you to the concept and implications of data privacy, including mechanisms and protocols that are used to preserve data privacy in practice. We will study concepts such as differential privacy, database anonymization, anonymous communication, and algorithmic fairness. We will also discuss privacy in the context of web privacy, social network privacy, human factors, and machine learning along with any policy implications.

Prerequisites: (COMP\_SCI 211-0 and COMP\_SCI 214-0) and (COMP\_SCI 262-0 or ELEC\_ENG 302-0 or IEMS 201-0 or IEMS 303-0 or STAT 210-0) or MS CS or PhDs CS or Instructor consent.

**COMP\_SCI 313-0 Tangible Interaction Design and Learning (1 Unit)** The use of tangible interaction to create innovative learning experiences, including distributed cognition, embodied interaction, cultural forms, and design frameworks. Prerequisites: COMP\_SCI 110-0 or COMP\_SCI 111-0.

#### **COMP\_SCI 314-0 Technology and Human Interaction (1 Unit)**

Understanding human interactions that occur both with and through technology; design, creation, and evaluation of technologies to support such interactions.

Prerequisite: Must be a Junior/Senior COMP\_SCI Major or Minor to enroll.

#### **COMP\_SCI 315-0 Design, Technology, and Research (1 Unit)**

Hands-on experience in the research learning environment. Students lead research projects in social and crowd computing, cyber-learning, human-computer interaction, and artificial intelligence.

Prerequisite: consent of instructor (by application only).

#### **COMP\_SCI 320-0 Proving Properties of Programs with Mechanized Logic (1 Unit)**

"This course will explore the state of the art in how to implement and prove facts about software. We will focus on small, functional programs and expressing properties of them via an (extremely) sophisticated dependent type system. By the end of this course, expect to have a better understanding of how to think about properties of programs, how to write code that demonstrates that those properties are true, and some experience programming in Agda.

Prerequisite: Students must have completed COMP\_SCI 212 and 321, be a MS CS Student and have completed CS 321, or be a CS PhD student to enroll."

#### **COMP\_SCI 321-0 Programming Languages (1 Unit)**

Introduction to key parts of programming languages: syntax, semantics, and pragmatics. Implementation of a series of interpreters that show how various aspects of programming languages behave.

Prerequisites: COMP\_SCI 111 and, COMP\_SCI 211, and COMP\_SCI 214 or COMP\_SCI MS or COMP\_SCI PhDs or Instructor permissioning.

#### **COMP\_SCI 322-0 Compiler Construction (1 Unit)**

The compiler is the programmer's primary tool. Understanding the compiler is therefore critical for programmers, even if they never build one. Furthermore, many design techniques that emerged in the context of compilers are useful for a range of other application areas. This course introduces students to the essential elements of building a compiler: parsing, context-sensitive property checking, code linearization, register allocation, etc. To take this course, students are expected to already understand how programming languages behave, to a fairly detailed degree. The material in the course builds on that knowledge via a series of semantics preserving transformations that start with a fairly high-level programming language and culminate in machine code.

Prerequisite: COMP\_SCI 213-0 and COM\_SCI 214-0 or consent of instructor.

#### **COMP\_SCI 323-0 Code Analysis and Transformation (1 Unit)**

Fast, highly sophisticated code analysis and code transformation tools are essential for modern software development. Before releasing its mobile apps, Facebook submits them to a tool called Infer that finds bugs by static analysis, i.e., without even having to run the code, and guides developers in fixing them. Google Chrome and Mozilla Firefox analyze and optimize JavaScript code to make browsers acceptably responsive. Performance-critical systems and application software would be impossible to build and evolve without compilers that derive highly optimized machine code from high-level source code that humans can understand. Understanding what modern code analysis and transformation techniques can and can't do is a prerequisite for research on both software engineering and computer architecture since hardware relies on software to realize its potential. In this class, you will learn the fundamentals of code analysis and transformation, and you will apply them by extending LLVM, a compiler framework now in production use by Apple, Adobe, Intel and other industrial and academic enterprises.

Prerequisites: COMP\_SCI 213-0 and COM\_SCI 214-0 or consent of instructor.

#### **COMP\_SCI 324-0 Dynamics of Programming Languages (1 Unit)**

The goal of this course is to introduce you to the semantics of programming languages (PLs). While CS 321 explains the meaning for different PL features, such as store and control, through interpreters, this course gives them meaning using simple math. And by simple, I mean as simple as high-school algebra. Hence, the course will help you build a foundational understanding of PLs by breaking them down to their most basic ingredients without having to go through PLAI or another meta language. In other words, the course will teach you how to construct mathematical models of PLs. Moreover, because of their mathematical and foundational nature, these models will enable you to describe precisely properties of PLs that in CS 321 we only talked about in an intuitive manner.

Prerequisite: CS 321 or CS PhD or Permission Instructor.

#### **COMP\_SCI 325-0 Artificial Intelligence Programming (1 Unit)**

Introduction to LISP and programming knowledge-based systems and interfaces. Strong emphasis on writing maintainable, extensible systems. Topics include semantic net-works, frames, pattern matching, deductive inference rules, case-based reasoning, and discrimination trees. Project-driven. Substantial programming assignments.

Prerequisite: COMP\_SCI 110-0, COMP\_SCI 111-0, or programming experience.

#### **COMP\_SCI 326-0 Introduction to the Data Science Pipeline (1 Unit)**

This course aims to cover various tools in the process of data science for obtaining, cleaning, visualizing, modeling, and interpreting data. Most of

the tools introduced in this course will be based on Python, although the idea can be applied to similar tools in other programming languages. As the outcome of this course, the students should be able to independently work on real-life datasets with large scales and gain insights from them. Prerequisites: CS 212 and CS 214 or COMP\_SCI MS or COMP\_SCI PhDs or Instructor consent.

#### **COMP\_SCI 327-0 Generative Methods (1 Unit)**

Generative Methods are algorithms which can be used to create. Programmers use grammars to make humorously chatty Twitterbots, Voronoi diagrams to create virtual cities and landscapes, and machine learned models to generate realistic oil portraits from selfies. This class will explore a range of different methods and tools, exposing you to the modern cutting edge of creative coding as you develop your own portfolio of JS/HTML apps.

Prerequisites: CS 111 and 150 or Instructor consent.

#### **COMP\_SCI 329-0 HCI Studio (1 Unit)**

Human-Computer Interaction (HCI) serves as the bridge between computing and humanity. In this class we will develop our critical thinking skills by learning effective structures for designing HCI systems. We will also soften into a deeper understanding of people's problems by developing our capacities for humility, empathy, and curiosity. Learning occurs through instructional activities, team projects, and studio critique. Prerequisite: CS 214 and has not completed CS 330 or Graduate Standing and has not completed CS 330.

#### **COMP\_SCI 330-0 Human Computer Interaction (1 Unit)**

Introduction to human-computer interaction and design of systems that work for people and their organizations. Understanding the manner in which humans interact with and use computers for productive work. For more detail - <https://www.mccormick.northwestern.edu/computer-science/academics/courses/descriptions/330-1.html>.

Prerequisite: CS 214 and has not completed CS329 or Graduate Standing and has not completed CS 329.

#### **COMP\_SCI 331-0 Introduction to Computational Photography (1 Unit)**

Fundamentals of digital imaging and modern camera architectures. Hands-on experience acquiring, characterizing, and manipulating data captured using a modern camera platform.

Prerequisite: COMP\_SCI 150 or COMP\_SCI 211 or Consent of Instructor.

#### **COMP\_SCI 332-0 Online Markets (1 Unit)**

Online markets are causing significant changes to society. Examples include eBay, airBnB, tinder, Uber, stackexchange, and Amazon. This class gives an introduction to the science of online markets combining topics from game theory and economics with topics from machine learning and algorithms. The two main topics of interest are how individuals in these market places optimize their strategies and how the market designer optimizes the rules of the market place so that, when individuals optimize their strategies, desired market outcomes are achieved. Student work will be a mix of problem sets and short projects.

Prerequisites: CS 212 (Discrete Math) and CS 214 (Data Structures) or CS 336 (Algorithms) or ECON 380-1 (Game Theory).

#### **COMP\_SCI 333-0 Interactive Information Visualization (1 Unit)**

This course covers theory and techniques for information visualization: the use of interactive interfaces to visualize abstract data. The course targets students interested in using visualization in their work or in building better visualization tools and systems. Students will learn to design and implement effective visualizations, critique others' visualizations, conduct exploratory visual analysis, and navigate research on information visualization.

Prerequisites: COMP\_SCI 214-0 or consent of instructor.

#### **COMP\_SCI 334-0 Introduction to Computational Linguistics (1 Unit)**

Hands-on introduction to computational methods in empirical linguistic analysis and natural language processing. Topics include language modeling, text classification, linguistic annotation, computational semantics, and machine translation. Students will implement and apply computational models to real linguistic datasets, and conclude the course with a final project.

Prerequisite: COMP\_SCI 110 or COMP\_SCI 111 or be a graduate COMP\_SCI student or have permission of Instructor.

#### **COMP\_SCI 335-0 Introduction to the Theory of Computation (1 Unit)**

Mathematical foundations of computation, including computability, relationships of time and space, and the P vs. NP problem.

Prerequisite: CS 212 or CS PhDs or consent of instructor.

#### **COMP\_SCI 336-0 Design & Analysis of Algorithms (1 Unit)**

Analysis techniques: solving recurrence equations. Algorithm design techniques: divide and conquer, the greedy method, backtracking, branch-and-bound, and dynamic programming. Sorting and selection algorithms, order statistics, heaps, and priority queues.

Prerequisite: CS 212 and CS 214, or CS PhDs or consent of instructor.

#### **COMP\_SCI 337-0 Natural Language Processing: Classical Approaches (1 Unit)**

Semantics-oriented introduction to natural language processing, broadly construed. Representation of meaning and knowledge inference in story understanding, script/frame theory, plans and plan recognition, counter-planning, and thematic structures.

Prerequisite: COMP\_SCI 348-0 or consent of instructor.

#### **COMP\_SCI 338-0 Practicum in Intelligent Information Systems (1 Unit)**

A practical excursion into building intelligent information systems. Students develop a working program in information access, management, capture, or retrieval. Project definition, data collection, technology selection, implementation, and project management.

#### **COMP\_SCI 339-0 Introduction to Database Systems (1 Unit)**

Data models and database design. Modeling the real world: structures, constraints, and operations. The entity relationship to data modeling (including network hierarchical and object-oriented), emphasis on the relational model. Use of existing database systems for the implementation of information systems.

Prerequisites: COMP\_SCI 214-0 and (COMP\_SCI 213-0 or COMP\_ENG 205-0) or CS Graduate Standing.

#### **COMP\_SCI 340-0 Introduction to Networking (1 Unit)**

A top-down exploration of networking using the five-layer model and the TCP/IP stack, covering each layer in depth. Students build web clients, servers, and a TCP implementation and implement routing algorithms.

Prerequisites: COMP\_SCI 214-0 and (COMP\_SCI 213-0 or COMP\_ENG 205-0).

#### **COMP\_SCI 341-0 Social Networks Analysis (1 Unit)**

The use of social network analysis to understand the growing connectivity and complexity in the world around us on different scales, ranging from small groups to the World 'Wide Web. How we create social, economic, and technological networks, and how they enable and constrain attitudes and behaviors. Taught with IEMS 341-0 and COMM\_ST 352-0; may not receive credit for the completion of both. Must be a COMP\_SCI Student to enroll.

**COMP\_SCI 341-SA Social Networks Analysis (1 Unit)** The use of social network analysis to understand the growing connectivity and complexity in the world around us on different scales, ranging from small groups to the World 'Wide Web. How we create social, economic, and technological networks, and how they enable and constrain attitudes and behaviors.

Restricted to students participation in Northwestern study abroad programs.

#### **COMP\_SCI 343-0 Operating Systems (1 Unit)**

Fundamental overview of operating systems, including: concurrency (processes, synchronization, semaphores, monitors, deadlock); memory management (segmentation, paging virtual memory policies); software system architectures (level structures, microkernels); file systems (directory structures, file organization, RAID); protection (access control, capabilities, encryption, signatures, authentication). Requires substantial programming projects.

Prerequisites: COMP\_SCI 214-0 and COMP\_SCI 213-0, or COMP\_SCI 214-0 and COMP\_ENG 205-0.

#### **COMP\_SCI 344-0 Design of Computer Problem Solvers (1 Unit)**

Principles and practice of organizing and building artificial intelligence reasoning systems. Pattern-directed rule systems, truth-maintenance systems, and constraint languages.

Prerequisites: COMP\_SCI 348-0 and COMP\_SCI 325-1 or equivalent LISP experience.

#### **COMP\_SCI 345-0 Distributed Systems (1 Unit)**

Basic principles behind distributed systems (collections of independent components that appear to users as a single coherent system) and main paradigms used to organize them.

Prerequisites: COMP\_SCI 213-0 and COMP\_SCI 214-0.

#### **COMP\_SCI 346-0 Microcontroller System Design (1 Unit)**

Structure and timing of typical microprocessors. Sample microprocessor families. Memories, UARTS, timer/counters, serial devices, and related devices. MUX and related control structures for building systems. Standard bus structures. Interrupt programming. Hardware/software design tradeoffs. For syllabus, please visit - <https://www.mccormick.northwestern.edu/computer-science/academics/courses/descriptions/346.html>.

Prerequisites: COMP\_SCI 211 and (COMP\_SCI 213 or COMP\_ENG 205).

**COMP\_SCI 347-0 Conversational AI (1 Unit)** Principles and practices of creating AI systems which interact with people through conversations.

This includes knowledge-rich natural language understanding, multimodal interactions (i.e. speech and sketching), principles of dialogue drawn from cognitive science, question-answering, and architectures for building conversational AI systems. Involves substantial programming and project work. Class sessions include both lectures and studio instruction. Prerequisites: COMP\_SCI 371 or permission of instructor.

#### **COMP\_SCI 348-0 Introduction to Artificial Intelligence (1 Unit)**

Core techniques and applications of AI. Representing, retrieving, and applying knowledge for problem solving. Hypothesis exploration. Theorem proving. Vision and neural networks.

Prerequisites: COMP\_SCI 111 and COMP\_SCI 214 or COMP\_SCI 111 and CogSci major or COMP\_SCI MS or COMP\_SCI PhDs or Instructor permission.

#### **COMP\_SCI 349-0 Machine Learning (1 Unit)**

Study of algorithms that improve through experience. Topics typically include Bayesian learning, decision trees, genetic algorithms, neural networks, Markov models, and reinforcement learning. Assignments include programming projects and written work.

Prerequisites: COMP\_SCI grad standing OR (COMP\_SCI 214 and (MATH 240-0 or GEN\_ENG 205-1 or GEN\_ENG 206-1) and (IEMS 201-0 or IEMS 303-0 or ELEC\_ENG 302-0 or STAT 210-0 or MATH 310-1).

#### **COMP\_SCI 350-0 Introduction to Computer Security (1 Unit)**

Basic principles and practices of computer and information security. Software, operating system, and network security techniques, with

detailed analysis of real-world examples. Topics include cryptography, authentication, software and operating system security (e.g., buffer overflow), Internet vulnerability (DoS attacks, viruses/worms, etc.), intrusion detection systems, firewalls, VPN, and web and wireless security.

Prerequisite: COMP\_SCI 213-0 or equivalent or consent of instructor; COMP\_SCI 340-0 highly recommended.

#### **COMP\_SCI 351-1 Introduction to Computer Graphics (1 Unit)**

Mathematical software and hardware requirements for computer graphics systems. Data structures and programming languages. Random displays. Graphic applications.

Prerequisite: COMP\_SCI 214-0 or Graduate standing.

#### **COMP\_SCI 351-2 Intermediate Computer Graphics (1 Unit)**

Methods and theory of computer graphics. Project-oriented approach. Describing shapes, movement, and lighting effects; interactive elements.

Prerequisites: COMP\_SCI 214-0 and COMP\_SCI 351-1 or Graduate standing.

#### **COMP\_SCI 352-0 Machine Perception of Music & Audio (1 Unit)**

Machine extraction of musical structure in audio and MIDI and score files, covering areas such as source separation and perceptual mapping of audio to machine-quantifiable measures.

Prerequisite: COMP\_SCI 211-0 and COMP\_SCI 214-0.

#### **COMP\_SCI 353-0 Natural & Artificial Vision (1 Unit)**

This course covers the mathematical operations underlying computer vision and their embodiment in naturally-occurring visual systems.

Students will learn basics of image processing, neuroscience, and computational imaging.

Prerequisite: (Math 228-1 or Math 230-1) and (Gen\_Eng 205-1 or Math 240) or Instructor Permission.

#### **COMP\_SCI 354-0 Computer System Security (1 Unit)**

The past decade has seen an explosion in the concern for the security of information. This course introduces students to the basic principles and practices of computer system and networking security, with detailed analysis of real-world examples and hands-on practice. Topics include the basic crypto, authentication, reverse engineering, buffer overflow attacks, vulnerability scanning, web attacks, firewalls, intrusion detection/prevention systems, etc. We will first introduce the basic theory for each type of attack; then we will actually carry them out in 'real-world' settings. The goal is to learn security by learning how to view your machine from a hacker's perspective. In addition, we encourage students to participate in the UCSB International Capture the Flag Competition. Capture the Flag is a network security exercise where the goal is to exploit other machines while defending your own. In fact, this course should prepare you for any one of many capture the flag competitions that take place year-round. We will learn about different types of hacks and perform them. After learning how to execute such exploits and penetrate a network, we will discuss ways to protect a network from others exploiting the same vulnerabilities. Understanding security is essential in all fields of software development and computing. For major or minors in Computer Science, this course can satisfy the system breadth.

Prerequisite: COMP\_SCI 211-0 and COMP\_SCI 213-0 or COMP\_SCI 211-0 and COMP\_ENG 205-0.

#### **COMP\_SCI 355-0 Digital Forensics and Incident Response (1 Unit)**

This course aims to teach students the concepts of Digital Forensics and Incident Response. The technical content taught in the class consists of deep knowledge of filesystems and operating systems so that students know which digital artifacts to investigate in data breach scenarios.

Labs and assignments are a sanitized version of real-world intrusions by nation-state actors and cybercriminals.

Prerequisites: COMP\_SCI 213-0 and COMP\_SCI 354-0 or COMP\_SCI 213-0 and COMP\_SCI 343-0.

#### **COMP\_SCI 358-0 Introduction to Parallel Computing (1 Unit)**

Introduction to parallel computing for scientists and engineers. Shared-memory parallel architectures and programming, distributed memory, message-passing data-parallel architectures, and programming.

Prerequisite: (CS 213 or (CE 205 & 211)) or MS CS/CE or PhDs CS/CE or permission of Instructor.

#### **COMP\_SCI 362-0 Foundations of Quantum Computing and Quantum Information (1 Unit)**

This course will be an introduction to the theory of quantum computation from a computer science perspective. Quantum computing holds great promise for obtaining substantial computational improvements over classical computing for many problems. In this course, we will cover the basics of quantum computation, and different topics that explore both the capabilities and limitations of quantum computers. Topics will include (subject to change) the basics of quantum information, quantum circuits, quantum algorithms, quantum complexity theory, quantum query complexity, and quantum communication complexity. No knowledge of quantum mechanics is required. We will cover the necessary physics concepts that are needed for this course.

Prerequisite: Undergrad CS 262 or CS 335 or CS 336.

#### **COMP\_SCI 367-0 Wireless and Mobile Health (mHealth) (1 Unit)**

Wireless and Mobile health (mHealth) aims to improve our health and well-being by utilizing data provided by technologies such as wearables, IoT devices, and mobile phones. This course will cover foundational knowledge and topical work in mHealth. Students will learn the essential steps needed to create a mHealth system from conception to evaluation through assigned reading, group discussions, and some hands-on projects. Since mHealth is an interdisciplinary field, we will cover a range of literature in human-computer interaction, wearable systems, machine learning, and health. Readings, hands-on projects, and in-class activities will prepare students for their final creative assignment that involves writing and presenting a research proposal for a mHealth system or research idea. Although this course will cover some technical aspects of mHealth, we encourage students with different backgrounds to enroll as they will work in interdisciplinary teams. <https://www.mccormick.northwestern.edu/computer-science/academics/courses/descriptions/397-497-14.html>.

#### **COMP\_SCI 368-0 Programming Massively Parallel Processors with CUDA (1 Unit)**

This course focuses on developing and optimizing applications software on massively parallel graphics processing units (GPUs). Such processing units routinely come with hundreds to thousands of cores per chip and sell for a few hundred to a few thousand dollars. The massive parallelism they offer allows applications to run 2x-450x faster than on conventional multicores. However, to reach this performance improvement, the application must fully utilize the computational, storage and communication resources provided by the device. This course discusses state-of-the-art parallel programming optimization methods to achieve this goal.

Prerequisites: COMP\_SCI 213 and (COMP\_SCI 211 or COMP\_SCI 230) or consent of instructor.

#### **COMP\_SCI 370-0 Computer Game Design (1 Unit)**

Plot, narrative, and character simulation for creating game worlds; artificial intelligence for synthetic characters; tuning gameplay. Substantial programming and project work.

Prerequisites: COMP\_SCI 214-0; 1 unit of COMP\_SCI 322-0, COMP\_SCI 343-0, COMP\_SCI 348-0, or COMP\_SCI 351-1, COMP\_SCI 351-2.

#### **COMP\_SCI 371-0 Knowledge Representation and Reasoning (1 Unit)**

Principles and practices of knowledge representation, including logics, ontologies, commonsense knowledge, and semantic web technologies.

Prerequisites: COMP\_SCI 348-0 or COMP\_SCI 325, or equivalent experience with artificial intelligence.

#### **COMP\_SCI 372-0 Designing and Constructing Models with Multi-Agent Languages (1 Unit)**

This course will begin with an introduction to the multi-agent language NetLogo. Students will design and implement several NetLogo models and analyze their behavioral regimes. Students will also learn to build models of interaction on social networks (or other types of networks). We will cover methodology for verifying, validating and replicating agent-based models and comparisons with systems dynamics and equation-based models. NetLogo comes with many extensions that support a variety of additional features. Students can use these extensions to create specialized models, such as complex networks, real-time data extraction, data mining, connections to physical devices, etc.. Students will also have an opportunity to explore existing and create their own participatory simulations using the HubNet architecture as well as exploring connecting real world sensors and motors to models. Students can also explore multi-level agent-based modeling in which hundreds or thousands of models are connected with NetLogo's LevelSpace extension. Prerequisite: Senior Comp\_sci or Senior Comp\_Eng or Consent of instructor.

#### **COMP\_SCI 374-0 Causal Graphical Models (1 Unit)**

We know that correlation does not imply causation, but careful analyses of correlations are often our only way to quantify cause and effect in domains ranging from healthcare to education. This course introduces causal inference methods, primarily using probabilistic graphical models, to identify and estimate counterfactual quantities as functions of observational data. We will discuss common challenges to causal inference, including confounding bias, missing data, measurement error, and selection bias. The final project will allow students to choose a dataset on which to perform a causal data analysis.

Prerequisite: Permission of Instructor.

#### **COMP\_SCI 375-0 Digital Musical Instrument Dsg (0 Unit)**

Digital Musical Instrument Design approaches Human Computer Interaction through the lens of artistic creation in sound. Its perspective sits at the intersection of design, engineering, musicianship, and the sonic arts. This course will introduce students to fundamentals of creating their own musical instruments using highly-accessible modern technologies in hardware and software. We will explore the influence and feedback between the tool (e.g. an instrument), the tool user (a musician/producer), the creative output (the music), as well as the audience. Along the way, we will learn about user interaction design, physical computing (using Arduino), sensor technologies, signal creation and interpretation (using Max or PureData), and musical expression. Students will create a new musical instrument by the end of the course and the final project will involve a performance using their instrument.

#### **COMP\_SCI 376-0 Computer Game Design and Development (1 Unit)**

Introduction to design of simulation-based media, with an emphasis on 2D game design. Mathematical preliminaries: linear, affine, and projective spaces, linear transforms, inner and exterior products, unit quaternions; Architecture: update/render loop, component systems, serialization and deserialization, event handling and asynchronous processing, multitasking; Rendering: scene graphs, meshes, shaders, sprites; Networking; Audio; Physics: particles, rigid bodies, collision detection; Gameplay design.

Prerequisite: COMP\_SCI 214-0.

#### **COMP\_SCI 377-0 Game Design Studio (1 Unit)**

In this course, students will design and develop games using the Unity game engine, with focus on team-based projects and agile development practices. Lectures will cover game design theory, game architecture and implementation, and the business of game development. Students will participate in class discussion and evaluation of projects in progress, to develop their skills in iterative design and implementation.

Prerequisite: COMP\_SCI 214 and COMP\_SCI 376-0.

**COMP\_SCI 380-0 Software Quality Engineering (1 Unit)**

This course introduces modern software quality engineering practices regarding both product and process quality assurance. For product quality, we explore topics such as taxonomy of testing techniques, test-driven development, unit testing and integration testing, evaluating test quality, and internationalization. For process quality, we introduce version control system, build tool, CI/CD pipeline, and their impact on software quality.

Prerequisite: CS 211-and CS 214 or MS CS or PhDs CS or PhD CS+LS or PhD TSB or permission of Instructor.

**COMP\_SCI 387-0 Responsible Software Engineering (1 Unit)**

As software's role in the world ever increases, so do the consequences of software going wrong—be it accidentally or maliciously. As software engineers, it is our responsibility to guard against such failures to the best of our ability, and to ensure that the software we build benefits the public rather than harms them. In this class, we will study a variety of software failures—both recent and historical—to understand what led to them and draw lessons from them. In parallel, we will explore techniques, tools, and processes that can help prevent such failures, and should thus be part of a responsible software engineer's toolbox.

Prerequisites: CS 213 and 214 or CS MS or CS PhDs or Instructor permission.

**COMP\_SCI 388-0 Software Engineering Beyond Programming (1 Unit)**

Many important issues and tradeoffs in software engineering appear only once projects reach a certain scale: large codebases, large teams, long periods of time. In this class, we will study a number of such issues, as well as practices and processes to help manage them.

Prerequisite: Application only.

*Registration is By Application Only*

**COMP\_SCI 392-0 Rapid Prototyping for Software Innovation (1 Unit)**

This is a course about developing working prototypes of full-stack mobile web software applications in rapid iterations. Teams design and implement three distinct applications over ten weeks. These projects are the context for introducing (1) cross-functional team development, (2) lean agile value-first product development, and (3) specific web application frameworks and development tools, such as React, Firebase, Cypress, and Github Actions for continuous integration.

Prerequisites: CS junior, senior, or graduate OR permission of the instructor. Familiarity with JavaScript and HTML is presumed.

**COMP\_SCI 393-0 Software Construction (1 Unit)**

Building software is a craft that requires careful design. This course teaches software design principles in a studio setting. Each week, students present their programs to the class for review. Together, the class evaluates the programs for correctness and, more importantly, clarity and design. Expect to learn how to build reliable, maintainable, extensible software and how to read others' codes.

Prerequisites: COMP\_SCI 211-0 and COMP\_SCI 214-0.

**COMP\_SCI 394-0 Agile Software Development (1 Unit)**

Developing mobile and web applications, using modern sustainable agile practices, such as backlogs, user stories, velocity charts, and test driven development, to deliver value as quickly as possible to end users, clients, developers, and the development organization.

Prerequisites: Consent of instructor.

**COMP\_SCI 396-0 Special Topics in Computer Science (1 Unit)**

Topics suggested by faculty and approved by the department. Equivalent to 397 but intended to apply toward courses for the computer science major. Students are not allowed to take the same topic multiple times.

**COMP\_SCI 397-0 Special Projects in Computer Science (1 Unit)**

Projects suggested by faculty and approved by the department. Equivalent to 396 but intended to apply toward courses for the computer science major and its project requirement.

**COMP\_SCI 398-0 Research Track Practicum (1 Unit)**

Research Track Practicum course, students will continue progress on their proposed research projects. This class can be count toward CS Project or CS Technical Electives.

**COMP\_SCI 399-0 Projects (1 Unit)** Seminar and projects for advanced undergraduates on subjects of current interest in electrical and computer engineering.