COMPUTER SCIENCE

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 130-0 Tools and Technology of the World-Wide Web (1 Unit) Introduction to the theory and practice of developing sites on and technology for the web. Basics of HTML, JavaScript, ASP, and CGI programming.

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. *Empirical and Deductive Reasoning Foundational Dis*

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 Mathematical Foundations of Comp Science (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. Prerequisite: (EECS 110 or EECS 111) and Math 228-1 or 230-0.

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).

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 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.

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. Prerequisite: COMP_SCI 110-0, COMP_SCI 111-0, or consent of instructor.

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.

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.

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 and 212 and 214 or Instructor permission - (Programming experience and familiarity with basics of discrete math and statistics/probability).

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. Prerequisite: COMP_SCI 110-0.

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.

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 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 Graduate standing.

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 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.
Prerequisite: COMP_SCI 213-0.

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.

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.

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: COMP_SCI 214-0 or Graduate Standing or Consent of instructor.

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.
Prerequisite: COMP_SCI 211-0 or Graduate standing or Consent of instructor.

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 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: COMP_SCI 212-0 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: COMP_SCI 111-0, COMP_SCI 212-0, or CS Graduate Standing or consent of instructor.

COMP_SCI 337-0 Natural Language Processing (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 Mechanism Design (1 Unit)
Applying algorithms and microeconomics to derive a theory of the design of mechanisms that produce desired outcomes despite counteractive inputs by outside agents. Key application areas: auctions, markets, networking protocols.

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, microkernals); 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 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 CS Graduate Standing.

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 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.

COMP_SCI 367-0 Wireless and Mobile Health: Passive Sensing Data Analytics (1 Unit)
A hands-on introduction and experience to the growing field of mobile Health. Students work together on a project with clinicians and faculty in medicine, building a unique mHealth system while testing their system on a small population. Theory-driven project hypothesis, technology

selection and development, passive sensing data analytic chain understanding and implementation, and project management.

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.
Prerequisite: COMP_SCI 348-0, COMP_SCI 325-1, 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.

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 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.

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