

Bachelor of Science in Computer Science

Bachelor of Arts in Computing and Information Systems

Note

Because Computer Science undergraduate degrees are offered both in McCormick and Weinberg, this document is structured differently from a typical McCormick (BS) or Weinberg (BA) curriculum. It describes the elements of the degree in a manner independent of the frameworks of both schools. This description is intended to explain the nature of the degree to students who may be unfamiliar with the frameworks, and to external readers. Appendix A then shows how the curriculum is mapped into the McCormick framework, while Appendix B shows how it is mapped into the Weinberg framework.

Students should consult with their advisors in case of ambiguity or special cases.

1. Philosophy

Computer Science as a field grew out of Electrical Engineering, Mathematics, and Psychology over 50 years ago. It synthesized aspects of these fields and grew exponentially over the past half century, both in terms of the number of Computer Science practitioners, and its economic and social impact on the world. The field continues its exponential growth. In November, 2005, the U.S. Bureau of Labor Statistics estimated that almost 60% of new science and engineering jobs and net replacements in the United States through 2014 would be for computer specialists.

Given this growth, and the sheer breadth and scale of the Computer Science enterprise in industry and academia, what are the goals of the Northwestern undergraduate degree? A Northwestern Computer Science graduate will

- Comprehend the breadth of Computer Science, its key intellectual divisions and questions, and its past and likely future impacts on engineering, science, medicine, business, and law;
- Approach problems from the algorithmic perspective, understanding the nature of and broad reach of computation and how to apply it abstractly;
- Approach problems from the systems perspective, understand the evolving layers of the software/hardware stack and how to use and extend them;
- Approach problems from the intelligence perspective, understanding how to make progress against seemingly intractable problems;
- Design and implement complex software systems, individually and as a team member; and

- Design and implement effective human-computer interfaces.

Additionally, Northwestern graduates also will have had the opportunity to broaden their education by taking advantage of Computer Science's strong connections to Northwestern programs in Computer Engineering, Cognitive Science, and the Learning Sciences. Northwestern graduates may have also participated in directed research.

A Northwestern graduate will be imminently employable in the computer and software industries, and well beyond, as skills such as these are widely sought after. Our program will also provide effective preparation for graduate studies in Computer Science.

2. Engineering or Liberal Arts

Northwestern offers Computer Science degrees within McCormick, the Engineering and Applied Sciences School, and Weinberg, the Arts and Sciences School. The Computer Science-specific elements/requirements of the two degrees are identical. The McCormick degree offers a background in engineering, while the Weinberg degree offers a background in liberal arts.

3. Prerequisite Graph and Schedule

A detailed prerequisite graph is available as a separate document, as is a typical schedule.

4. Components of the Curriculum

The Northwestern Computer Science Degree is composed of five distinct sets of requirements. Background requirements build up the student's engineering skills. The Core requirements represent essential knowledge for all computer scientists. The Breadth requirements provide exposure to every critical subfield of Computer Science. The Depth requirements provide the student with the opportunity to learn about two specializations in depth, leading to a project, and perhaps graduate courses and research. The Project requirement gives the student the experience of designing and building a complex software artifact.

4.1 Background

Background courses are those courses that fulfill the general requirements of the University and the School (i.e., McCormick or Weinberg), as well as non-CS courses that the faculty believe are foundational for or helpful for understanding Computer Science.

Because background courses are strongly dependent on the School, in this section we present the CS background requirements at a high level. Appendices A and B give detailed descriptions of required or recommended courses within the two Schools. We require students to take courses that teach the following:

- **Continuous Mathematics.** Students must learn univariate differential and integral calculus, multivariate differential calculus, and linear algebra.
- **Probability and Statistics.** Students must learn basic probability theory, and basic statistics, including descriptive statistics and hypothesis testing.
- **Physical and Life Sciences.** Students must meet the requirements of their school. We recommend that students choose physics and biology courses.
- **Social Sciences and the Humanities.** Students should acquire a firm grounding in these areas, ideally choosing courses that integrate into a theme.
- **Communication Skills.** Students must learn the fundamentals of effective written, graphical, and oral communication.

Students who have never programmed before, in any language, are encouraged to take one of the following courses as part of their background:

- EECS 110 (C) – An Introduction to Programming for Non-majors using the C programming language. C is a widely used systems programming language.
- EECS 110 (Python) – An Introduction to Programming for Non-majors using the Python programming language. Python is a widely used scripting language.

Students are generally encouraged to take the Python option.

4.2 Core

The core courses reflect what the faculty expect every graduate to know. The core courses, all of which are required, consist of:

- EECS 101 – An Introduction to Computer Science For Everyone. This “immigration course” describes the entire field of Computer Science.
- EECS 111 – Fundamentals of Computer Programming I. This course teaches principles and practices of computer programming using a functional programming language.
- EECS 211 – Fundamentals of Computer Programming II. This course teaches further principles and practices of computer programming using an imperative, object-oriented programming language.
- EECS 213 – Introduction to Computer Systems. This course teaches how the computer system works, from the level of transistors to the level of distributed systems, with a particular focus on instruction set architecture, compilers, and operating systems.
- EECS 310 – Discrete Mathematics. This course introduces the mathematics underlying much of Computer Science.
- EECS 311 – Introduction to Data Structures. This course introduces software structures and algorithms for storing, accessing, and transforming information and their implementation.

4.3 Breadth (Essential Areas)

The breadth courses reflect the areas of Computer Science that the faculty believe you should be exposed to. A student must take one course in each of the following areas. Each course can only count for a single area.

It is possible to use EECS 399 (independent study) courses within breadth, with advisor and committee approval. This is rare and requires a petition.

Theory: Theory courses study the nature of computation, the nature of computational problems, and the design of algorithms for solving these problems efficiently. The following courses are considered appropriate for satisfying the theory breadth requirement. The courses with asterisks are especially recommended.

- * EECS 336 – Design and Analysis of Algorithms
- * MATH 374 – Theory of Computability and Turing Machines
- EECS 328 – Numerical Methods
- EECS 356 – Formal Specification and Verification

Systems: Systems courses study the layers of the hardware/software stack, and the design, implementation, and evaluation of complex software systems, including computer security. The following courses are appropriate for satisfying the systems breadth requirement. The courses with asterisks are especially recommended.

- * EECS 322 – Compiler Construction
- * EECS 339 – Introduction to Databases
- * EECS 340 – Introduction to Networking
- * EECS 343 – Operating Systems
- EECS 303 – Digital Logic Design
- EECS 345 – Distributed Systems
- EECS 346 – Microprocessor Systems Design
- EECS 350 – Introduction to Security
- EECS 358 – Parallel Systems
- EECS 361 – Computer Architecture
- EECS 397 – Real-time Systems
- EECS 440 – Advanced Networking
- EECS 441 – Resource Virtualization
- EECS 442 – Dynamic Behavior of Applications, Hosts, and Networks
- EECS 443 – Advanced Operating Systems
- EECS 450 – Internet Security
- EECS 464 – Advanced Databases

Artificial Intelligence: Those in the field of Artificial Intelligence seek scientific understanding of the mechanisms underlying thought and intelligent behavior and look to embody these mechanisms in machines. Intelligent learning environments, computer games, and music retrieval systems are some applications of AI technology. The

following courses are appropriate for satisfying the AI breadth requirement. The courses with asterisks are especially recommended.

- * EECS 325 – AI Programming
- * EECS 337 – Semantic Information Processing
- * EECS 344 – Design of Computer Problem Solvers
- * EECS 348 – Introduction to AI
- * EECS 349 – Machine Learning
- EECS 360 – Models with Multi-agent Languages
- EECS 395/495 – AI For Interactive Entertainment
- EECS 395/495 – Knowledge Representation
- EECS 395/495 – Behavior-based Robotics

Interfaces: Courses in this area study the human-computer interface, including computer graphics and multimedia processing. The following courses are appropriate for satisfying the interface breadth requirement. The courses with asterisks are especially recommended.

- * EECS 330 – Human-Computer Interaction
- * EECS 351 – Introduction to Computer Graphics
- * EECS 352 – Machine Perception of Music
- * EECS 370 – Computer Game Design
- EECS 332 – Digital Image Analysis
- EECS 395 – Intermediate Computer Graphics
- EECS 395 – Advanced Computer Graphics
- EECS 395/495 – Computer Animation
- EECS 395/495 – Graphics and Perception
- EECS 395/495 – Image-based Modeling and Rendering
- EECS 395/495 – Human-centered Product Design

Software Development: Courses in this area provide opportunities to experience larger-scale software development in teams. Scaling software development is an important challenge. The following courses are appropriate for satisfying the software development breadth requirement. The courses with asterisks are especially recommended.

- * EECS 338 – Practicum in Intelligent Information Systems
- * EECS 394-1,2 – Software Project Management
- * EECS 395 – RTFM courses (<http://rtfm.cs.northwestern.edu>)

4.4 Depth

Students are expected to acquire depth in two of the areas listed below. Students should consult with their advisor to choose the two most appropriate areas of study for them. A total of six courses should be taken in the two chosen areas, with three courses in each area.

A student may petition to be allowed to focus on a single area, if desired. This petition needs to be approved by the student's advisor, but we intend for this to be straightforward.

A student may petition to use independent study courses (EECS 399s) in depth areas that permit them. This petition needs to be approved by the student's advisor and the committee. Again, we intend for this to be straightforward.

A student interested in defining his/her own depth area is encouraged to provide a justification for the proposed set of depth courses to his/her advisor, who may in turn petition the Computer Science Curriculum Committee for its approval.

The Computer Science Curriculum Committee will meet once per year, in the fall quarter, to consider additions, deletions, or changes to the depth areas and the list of appropriate courses for each area.

Theory: Theory courses study the nature of computation, the nature of computational problems, and the design of algorithms for solving these problems efficiently. The following courses are considered appropriate for satisfying the theory depth area. The courses with asterisks are especially recommended.

- * EECS 336 – Design and Analysis of Algorithms
- * MATH 374 – Theory of Computability and Turing Machines
- * EECS 395/495 – Current Topics in Algorithms
- * EECS 457 – Advanced Algorithms
- EECS 328 – Numerical Methods
- EECS 356 – Formal Specification and Verification
- EECS 357 – Introduction to VLSI CAD
- EECS 395/495 – Algorithms for Bioinformatics
- EECS 395/495 – Algorithmic Research for e-Commerce
- EECS 459 – VLSI Algorithmics
- EECS 399 – Independent Study

Systems: In the depth area of systems, a student will learn about the issues and principles involved in the design, implementation, measurement, and analysis of the complex software and software/hardware systems upon which applications are built. Students will learn how computer systems work, from the level of the hardware to the level of

worldwide distributed communication and computation. The principles and issues involved have broad utility, and the skills that student will learn are in high demand.

- * EECS 322 – Compiler Construction
- * EECS 339 – Introduction to Databases
- * EECS 340 – Introduction to Networking
- * EECS 343 – Operating Systems
- * EECS 361 – Computer Architecture
- EECS 345 – Distributed Systems
- EECS 350 – Introduction to Security
- EECS 358 – Parallel Systems
- EECS 395 – Appropriate Selected Topics - with advisor approval
- EECS 397 – Real-time Systems
- EECS 440 – Advanced Networking
- EECS 441 – Resource Virtualization
- EECS 442 – Dynamic Behavior of Applications, Hosts, and Networks
- EECS 443 – Advanced Operating Systems
- EECS 450 – Internet Security
- EECS 464 – Advanced Databases
- EECS 399 – Independent Study

Artificial Intelligence: Those in the field of Artificial Intelligence seek scientific understanding of the mechanisms underlying thought and intelligent behavior and look to embody these mechanisms in machines. Intelligent learning environments, computer games, and music retrieval systems are some applications of AI technology. The following courses are appropriate for depth in AI. The courses with asterisks are especially recommended.

- * EECS 325 – AI Programming
- * EECS 337 – Semantic Information Processing
- * EECS 344 – Design of Computer Problem Solvers
- * EECS 348 – Introduction to AI
- * EECS 349 – Machine Learning
- EECS 360 – Models with Multi-agent Languages
- EECS 395/495 – AI For Interactive Entertainment
- EECS 395/495 – Knowledge Representation
- EECS 395/495 – Behavior-based Robotics
- EECS 399 – Independent Study

Interfaces: Courses in this area study the human-computer interface, including computer graphics and multimedia processing. The following courses are appropriate for satisfying depth in interfaces. The courses with asterisks are especially recommended.

- * EECS 330 – Human-Computer Interaction

- * EECS 351 – Introduction to Computer Graphics
- * EECS 352 – Machine Perception of Music
- * EECS 370 – Computer Game Design
- EECS 332 – Digital Image Analysis
- EECS 395 – Intermediate Computer Graphics
- EECS 395 – Advanced Computer Graphics
- EECS 395/495 – Computer Animation
- EECS 395/495 – Graphics and Perception
- EECS 395/495 – Image-based Modeling and Rendering
- EECS 395/495 – Human-centered Product Design
- EECS 399 – Independent Study

Security: This area focuses on the fundamental principles of computer and communication security as well as their implications and applications in various domains of information technology. The following courses are appropriate for satisfying the security depth area. The courses with asterisks are especially recommended.

- * EECS 350 – Introduction to Computer Security
- * EECS 450 – Internet Security
- * EECS 510-4 Computer Security and Information Assurance
- EECS 322 – Compiler Construction
- EECS 339 – Introduction to Database Systems
- EECS 340 – Introduction to Networking
- EECS 343 – Operating Systems
- EECS 345 – Distributed Systems
- EECS 395 – Appropriate Selected Topics - with advisor approval
- EECS 440 – Advanced Networks
- EECS 441 – Resource Virtualization
- EECS 443 – Advanced Operating Systems
- EECS 399 – Independent Study

4.5 Project

Students must complete at least two quarters of project-oriented classes. The best option is a two-quarter independent study project (EECS 399) with a faculty member. However, it is also possible to satisfy the requirement by taking two courses with extensive project work. Students must seek the formal approval of their advisor for the option they choose.

Possible ways of satisfying the Project requirement include:

- A single two quarter EECS 399 project
- Two independent single quarter EECS 399 projects
- One EECS 399 project and one project course

- Two project courses
- Project courses are those courses listed on the CS Project Course List, which is available on the department's web site. The list is updated yearly by the CS Curriculum Committee.

4.6 Restrictions

Courses may not be double counted within the major program.

Students may take EECS 399 and EECS 338 no more than a total of four times. A petition is required for taking these courses additional times.

Appendix A. Mapping to the McCormick Framework

A.1 Mathematics: Students must take the following four courses.

- MATH 220 – Calculus I
- MATH 224 – Calculus II
- MATH 230 – Calculus III
- EECS 310 – Discrete Mathematics

A.2 Engineering Analysis: Students must take the following four courses.

- GEN_ENG 205-1 – EA-1 (Linear Algebra)
- GEN_ENG 205-2 – EA-2 (Mechanics)
- GEN_ENG 205-3 – EA-3 (Dynamics and Differential Equations)
- EECS 111 – Fundamentals of Computer Programming I

Note that students do not need to take EA-4 for the CS degree. However, students may elect to do so if they are also interested in other degrees in McCormick.

A.3 Engineering Design and Communication: Students must take the following three courses.

- IDEA 106-1 and ENGLISH 106-1 – EDC-1
- IDEA 106-2 and ENGLISH 106-2 – EDC-2
- GEN_CMN 102 Public Speaking

A.4 Basic Sciences: Students must take four courses that satisfy McCormick requirements. We recommend that students focus on Physics and Biology courses for maximum utility in their Computer Science Degree. We recommend that students take all of the following courses:

- PHYSICS 135-2 General Physics: Electromagnetics
- PHYSICS 135-3 General Physics: Wave Phenomena
- PHYSICS 335 Modern Physics

and choose one of these courses:

- BIOL SCI 210-1 Genetics and Evolutionary Biology
- CHEM_ENG 275 Molecular and Cell Biology for Engineers

Note that PHYSICS 135-1 – Mechanics is not suitable for satisfying the basic sciences requirement in engineering. However, the material in it is covered in EA-2. A common sequence for satisfying the basic sciences requirement in Computer Science is EA-2, PHYSICS 135-2, PHYSICS 135-3, PHYSICS 335, and BIOL SCI 210-1. This sequence will give you a firm grounding in classical and modern physics, upon which present and

possibly future computers are based, and in genetics and evolution, where computational approaches to biology and biological approaches to computation abound.

A.5 Basic Engineering: Students must take five courses that meet the following requirements.

- EECS 211 – Fundamentals of Computer Programming II must be taken. This course is in the area “Computer Programming”
- EECS 302 – Probabilistic Systems and Random Signals, IEMS 201 – Introduction to Statistics, or IEMS 303 – Statistics I must be taken. These courses are in the area “Probability, Statistics, and Process Control”.
- Three additional courses from McCormick’s Basic Engineering List (3 courses from at least two areas, excluding those covered by EECS 211 and 302). We recommend choosing from the following courses to meet this requirement:
 - EECS 202 – Introduction to Electrical Engineering (Area “Electrical Science”)
 - EECS 203 – Introduction to Computer Engineering (Area “Computer Architecture and Numerical Methods”)
 - EECS 222 – Fundamentals of Signals and Systems (Area “Electrical Science”)
 - EECS 328 – Numerical Methods for Engineers (Area “Computer Architecture and Numerical Methods”)

A.6. Social Sciences and Humanities: Students must take seven courses that meet the theme requirements of McCormick.

A.7 Unrestricted Electives: Students must take five courses.

Students who have never programmed before, in any language, are encouraged to use one of the unrestricted electives to take one of the following courses before taking EECS 111:

- EECS 110 (C) – An Introduction to Programming for Non-majors using the C programming language. C is a widely used systems programming language.
- EECS 110 (Python) – An Introduction to Programming for Non-majors using the Python programming language. Python is a widely used scripting language.

Students are generally encouraged to take the Python option. Note that EECS 110 is not a course within the CS major program.

A.8 Major Program (16) : Students must take 16 courses that meet the following requirements. No courses may be taken pass/fail. No courses may be double-counted within the major program.

A.8.1 Core Courses (3): Students must take the following three core courses as part of the major program. The core courses, including those mapped outside the major program, are described in detail in Section 4.2.

- EECS 101 – An Introduction to Computer Science For Everyone. This course should ideally be taken in the spring quarter of the freshman year
- EECS 213 – Introduction to Computer Systems
- EECS 311 – Introduction to Data Structures

A.8.2 Breadth Courses (5): Students must take five courses to satisfy the Breadth requirement, one in each area described in Section 4.3.

A.8.3 Depth Courses (6): Students must take six courses to satisfy the Depth requirement as described in Section 4.4.

A.8.4 Project Courses (2): Students must take two courses that satisfy the Project component of the curriculum as described in Section 4.5.

Appendix B. Mapping to the Weinberg Framework

B.1 Requirements to be met outside of the major program.

- Mathematics: MATH 220, 224, 230, and 240 are required
- Probability and Statistics: STATS 210 or MATH 310 are required
- Physical and Life Sciences: Students must satisfy the Natural Sciences distribution requirement. We recommend choosing from courses such as PHYSICS 135-1, 135-2, 135-3, 335; and BIOL SCI 210-1
- Social Sciences and Humanities: Covered by Weinberg's distributional requirements
- Communication Skills: Covered by Weinberg's distributional requirements

Students who have never programmed before, in any language, are encouraged to take one of the following courses before taking EECS 111:

- EECS 110 (C) – An Introduction to Programming for Non-majors using the C programming language. C is a widely used systems programming language.
- EECS 110 (Python) – An Introduction to Programming for Non-majors using the Python programming language. Python is a widely used scripting language.

Students who are generally encouraged to take the Python option.

B.2 Requirements to be met within the major program (19). No courses may be taken pass/fail. No courses may be double-counted within the major program.

B.2.1 Core Courses (6): Students must take the following six core courses as part of the major program. The core courses are described in detail in Section 4.2.

- EECS 101 – An Introduction to Computer Science For Everyone. This course should ideally be taken in the spring quarter of the freshman year
- EECS 111 – Fundamentals of Computer Programming I
- EECS 211 – Fundamentals of Computer Programming II
- EECS 213 – Introduction to Computer Systems
- EECS 310 – Discrete Mathematics
- EECS 311 – Introduction to Data Structures

B.2.2 Breadth Courses (5): Students must take five courses to satisfy the Breadth requirement, one in each area described in Section 4.3.

B.2.3 Depth Courses (6): Students must take six courses to satisfy the Depth requirement as described in Section 4.4.

B.2.4 Project Courses (2): Students must take two courses that satisfy the Project component of the curriculum as described in Section 4.5.