COMPUTER SCIENCE (CS)

CS 100
Introduction to the Profession
An introduction to science and engineering as a profession. Examines the problem-solving process used in engineering and science. Emphasizes the interdisciplinary and international nature of problem-solving and the need to evaluate solutions in terms of a variety of constraints: computational, financial, and social.
Lecture: 1 Lab: 2 Credits: 2
Satisfies: Communications (C)

CS 104
Introduction to Computer Programming for Engineers
Introduces the use of high-level programming language as a problem-solving tool in engineering including basic data structures and algorithms, structured programming techniques, and software documentation. Designed for students who have had little or no prior experience with computer programming. Students should only take one of these courses (CS 104, CS 105, CS 110, CS 115).
Lecture: 2 Lab: 1 Credits: 2

CS 105
Introduction to Computer Programming
Introduces the use of high-level programming language as a problem-solving tool, including basic data structures and algorithms, structured programming techniques, and software documentation. Designed for students who have had little or no prior experience with computer programming. Students should only take one of these courses (CS 104, CS 105, CS 110, CS 115).
Lecture: 2 Lab: 1 Credits: 2

CS 110
Computing Principles
An introduction to the following "big ideas" of computer science: (1) computing is a creative activity; (2) abstraction reduces information and detail to facilitate focus on relevant concepts; (3) data and information facilitate the creation of knowledge; (4) algorithms are used to develop and express solutions to computational problems; (5) programming enables problem solving, human expression, and creation of knowledge; (6) the internet pervades modern computing; and (7) computing has global impacts. Students should only take one of these courses (CS 104, CS 105, CS 110, CS 115).
Lecture: 2 Lab: 1 Credits: 2

CS 115
Object-Oriented Programming I
Introduces the use of a high-level object-oriented programming language as a problem-solving tool, including basic data structures and algorithms, object-oriented programming techniques, and software documentation. Designed for students who have had little or no prior experience with computer programming. For students in CS and CS-related degree programs. Students should only take one of these courses (CS 104, CS 105, CS 110, CS 115).
Lecture: 2 Lab: 1 Credits: 2

CS 116
Object-Oriented Programming II
Introduces more advanced elements of object-oriented programming, including dynamic data structures, recursion, searching and sorting, and advanced object-oriented programming techniques. For students in CS and CS-related degree programs.
Prerequisite(s): CS 115 with min. grade of C
Lecture: 2 Lab: 1 Credits: 2

CS 201
Accelerated Introduction to Computer Science
Problem-solving and design using an object-oriented programming language. Introduces a variety of problem-solving techniques, algorithms, and data structures in object-oriented programming.
Prerequisite(s): CS 104 with min. grade of C or CS 105 with min. grade of C or CS 110 with min. grade of C or CS 115 with min. grade of C
Lecture: 3 Lab: 2 Credits: 4

CS 330
Discrete Structures
Introduction to the use of formal mathematical structures to represent problems and computational processes. Topics covered include Boolean algebra, first-order logic, recursive structures, graphs, and abstract language models. Credit will not be granted for both CS 330 and MATH 230.
Prerequisite(s): CS 201 or CS 116
Lecture: 3 Lab: 1 Credits: 3

CS 331
Data Structures and Algorithms
Implementation and application of the essential data structures used in computer science. Analysis of basic sorting and searching algorithms and their relationship to these data structures. Particular emphasis is given to the use of object-oriented design and data abstraction in the creation and application of data structures.
Prerequisite(s): CS 116 or CS 201
Lecture: 3 Lab: 1 Credits: 3

CS 340
Programming Paradigms and Patterns
This class balances the imperative, object-oriented bent of the introductory programming sequence by presenting alternative programming paradigms and asks students to write complex programs from scratch while amassing a repertoire of reusable programming patterns and techniques. Programming assignments, drawn from various domains of computer science, will highlight the importance of selecting appropriate data structures, algorithms, and techniques for the problem at hand.
Prerequisite(s): CS 116 or CS 201
Lecture: 3 Lab: 0 Credits: 3
CS 350
Computer Organization and Assembly Language Programming
Introduction to the internal architecture of computer systems, including micro-, mini-, and mainframe computer architectures. Focuses on the relationship among a computer’s hardware, its native instruction set, and the implementation of high-level languages on that machine. Uses a set of assembly language programming exercises to explore and analyze a microcomputer architecture. Credit will not be granted for both CS 350 and ECE 242.
Prerequisite(s): CS 116* or CS 201*, An asterisk (*) designates a course which may be taken concurrently.
Lecture: 3 Lab: 1 Credits: 3
Satisfies: Communications (C)

CS 351
Systems Programming
Examines the components of sophisticated multilayer software systems, including device drivers, systems software, applications interfaces, and user interfaces. Explores the design and development of interrupt-driven and event-driven software.
Prerequisite(s): (CS 331 and CS 350) or (CS 331 and ECE 242)
Lecture: 3 Lab: 1 Credits: 3

CS 397
Special Projects
**Instructor permission required.**
Credit: Variable

CS 401
Introduction to Advanced Studies I
First course in a two-course sequence that is designed to prepare students for graduate study in computer science. Explores the implementation and application of fundamental data structures and algorithms with an emphasis on object-oriented programming in Java. Examines the relationship between these elements and the mathematical structures that form the foundation of computer science. This course does not apply toward M. S./Ph. D. credit in Computer Science.
Prerequisite(s): CS 200 or CS 201
Lecture: 2 Lab: 2 Credits: 3

CS 402
Introduction to Advanced Studies II
Second course in a two-course sequence that is designed to prepare students for graduate study in computer science. Explores the development of the multiple layers of software that form a sophisticated software system, from device drivers to application interfaces to user interfaces. Examines how computer architecture influences software development. Emphasizes the design and implementation of interrupt-driven/event-driven software.
Prerequisite(s): CS 401
Lecture: 2 Lab: 2 Credits: 3
CS 430  
Introduction to Algorithms  
Introduction to the design, behavior, and analysis of computer algorithms. Searching, sorting, and combinatorial algorithms are emphasized. Worst case, amortized, and expected bounds on time and space usage.  
Prerequisite(s): (CS 331 and CS 330) or (CS 331 and MATH 230) or CS 401 or CS 403  
Lecture: 3 Lab: 1 Credits: 3  
Satisfies: CS Technical Elective (T), Communications (C)

CS 440  
Programming Languages and Translators  
Study of commonly used computer programming languages with an emphasis on precision of definition and facility in use. Scanning, parsing, and introduction to compiler design. Use of compiler generating tools.  
Prerequisite(s): (CS 330 and CS 331) or (MATH 230 and CS 331) or CS 401 or CS 403  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)

CS 442  
Mobile Applications Development  
Students will learn a variety of software engineering techniques and design patterns to assist in the rapid development and prototyping of applications, leveraging frameworks and APIs provided by current mobile development platforms (such as Android and iOS). Application lifecycles, data management and persistence mechanisms, and user interface design, among other topics, will be covered. Industry speakers will be invited to speak about best practices. Students (individually or in teams) will take ideas from concept to final implementation and will present their work at the end of the semester. When appropriate, students may take the additional step of deploying their work on the appropriate application marketplace(s).  
Prerequisite(s): (CS 331 or CS 401) and (CS 351* or CS 402*), An asterisk (*) designates a course which may be taken concurrently.  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)

CS 443  
Compiler Construction  
This course covers the design and implementation of a compiler for modern languages by implementing the following: abstract syntax trees; intermediate representations; static analysis; fix-point operations; symbol tables and type checking; and first-order and high-order function implementation. Students will incrementally create a series of compilers.  
Prerequisite(s): CS 440  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)

CS 445  
Object Oriented Design and Programming  
Introduction to methodologies for object-oriented design and programming. Examines the object model and how it is realized in various object-oriented languages. Focuses on methods for developing and implementing object-oriented systems.  
Prerequisite(s): CS 331 or CS 401 or CS 403  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)

CS 447  
Distributed Objects  
This course provides an introduction to architecture, analysis, design, and implementation of distributed, multi-tier applications using distributed object technology. The course focuses on the services and facilities provided by an Object Request Broker (ORB). Students will use a commercially available ORB and Database Management System to develop distributed object applications.  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: Communications (C), CS Technical Elective (T)

CS 450  
Operating Systems  
Introduction to operating system concepts-including system organization for uniprocessors and multiprocessors, scheduling algorithms, process management, deadlocks, paging and segmentation, files and protection, and process coordination and communication.  
Prerequisite(s): CS 351 or (CS 401 and CS 402) or CS 403  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)

CS 451  
Introduction to Parallel and Distributed Computing  
This course covers general introductory concepts in the design and implementation of parallel and distributed systems covering all the major branches such as cloud computing, grid computing, cluster computing, supercomputing, and many-core computing.  
Prerequisite(s): CS 450  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)

CS 455  
Data Communications  
Introduction to data communication concepts and facilities with an emphasis on protocols and interface specifications. Focuses on the lower four layers of the ISO-OSI reference model.  
Prerequisite(s): CS 450  
Lecture: 3 Lab: 0 Credits: 3  
Satisfies: CS Technical Elective (T)
CS 456
Introduction to Wireless Networks and Performance
This class provides an opportunity for students to obtain a fundamental understanding of the nature and operation of the full range of wireless networks (personal, local area, wide area, and satellite) and their performance characteristics, future potential, and challenges through class lectures, assigned readings, homework, projects, and various hands-on experiences.
Prerequisite(s): CS 350 or ECE 242 or (CS 401 and CS 402) or CS 403
Lecture: 3 Lab: 0 Credits: 3
Satisfies: CS Technical Elective (T)

CS 458
Introduction to Information Security
An introduction to the fundamentals of computer and information security. This course focuses on algorithms and techniques used to defend against malicious software. Topics include an introduction to encryption systems, operating system security, database security, network security, system threats, and risk avoidance procedures.
Prerequisite(s): CS 425 or CS 450 or CS 455
Lecture: 3 Lab: 0 Credits: 3
Satisfies: Communications (C), CS Technical Elective (T)

CS 470
Computer Architecture
Introduction to the functional elements and structures of digital computers. Detailed study of specific machines at the register transfer level illustrates arithmetic, memory, I/O and instruction processing.
Prerequisite(s): (ECE 218 and CS 350) or (ECE 218 and ECE 242)
Lecture: 2 Lab: 2 Credits: 3
Satisfies: Communications (C), CS Technical Elective (T)

CS 480
Artificial Intelligence Planning and Control
Introduction to computational methods for intelligent control of autonomous agents, and the use of programming paradigms that support development of flexible and reactive systems. These include heuristic search, knowledge representation, constraint satisfaction, probabilistic reasoning, decision-theoretic control, and sensor interpretation. Particular focus will be places on real-world application of the material.
Prerequisite(s): (CS 331 and MATH 474*) or (CS 401 and CS 402), An asterisk (*) designates a course which may be taken concurrently.
Lecture: 3 Lab: 0 Credits: 3
Satisfies: CS Technical Elective (T)

CS 481
Artificial Intelligence Language Understanding
Theory and programming paradigms that enable systems to understand human language texts and extract useful information and knowledge. For example, extraction of structured event representations from news stories or discovering new research hypotheses by analyzing thousands of medical research articles. the course covers a variety of text analysis and text mining methods, with an emphasis on building working systems. Connections to information retrieval, data mining, and speech recognition will be discussed.
Prerequisite(s): (CS 331 or CS 401 or CS 403) and MATH 474
Lecture: 3 Lab: 0 Credits: 3
Satisfies: CS Technical Elective (T)

CS 482
Information and Knowledge Management Systems
This capstone course is designed as a project course whose purpose is to enable students to see how the various algorithms and systems they have learned about in their prerequisite courses can be used in context to create useful knowledge management tools. Class periods will be divided among discussion of design of information and knowledge management systems, lectures on effective project management techniques, and hands-on advising of student project group meetings.
Prerequisite(s): (CS 425 and CS 422 and CS 429) or (CS 425 and CS 422 and CS 481) or (CS 425 and CS 429 and CS 481)
Lecture: 3 Lab: 0 Credits: 3
Satisfies: Communications (C), CS Technical Elective (T)

CS 485
Computers and Society
Discussion of the impact of computer technology on present and future society. Historical development of the computer. Social issues raised by cybernetics.
Prerequisite(s): COM 421 or COM 424 or COM 425 or COM 428 or COM 435
Lecture: 3 Lab: 0 Credits: 3
Satisfies: Communications (C)

CS 487
Software Engineering I
Study of the principles and practices of software engineering. Topics include software quality concepts, process models, software requirements analysis, design methodologies, software testing and software maintenance. Hands-on experience building a software system using the waterfall life cycle model. Students work in teams to develop all life cycle deliverables: requirements document, specification and design documents, system code, test plan, and user manuals.
Prerequisite(s): (CS 331 or CS 401 or CS 403) and CS 425
Lecture: 3 Lab: 0 Credits: 3
Satisfies: CS Technical Elective (T), Communications (C)

CS 491
Undergraduate Research
**Instructor permission required.**
Credit: Variable

CS 492
Introduction to Computer Science Research
Prepares undergraduate computer science majors for conducting research.
Lecture: 1 Lab: 0 Credits: 1

CS 495
Topics in Computer Science
This course will treat a specific topic, varying from semester to semester, in which there is particular student or staff interest.
Credit: Variable

CS 497
Special Projects
Special projects.
Credit: Variable