Computer Science & Engineering

About Computer Science & Engineering

The field of computer science and engineering studies the design, analysis, implementation and application of computation and computer technology. Computing plays an important role in virtually all fields, including science, medicine, music, art, business, law and human communication; hence, the study of computer science and engineering can be interdisciplinary in nature. Whether a student's goal is to become a practitioner or to take a few courses to develop a basic understanding of computing for application to another field, the Department of Computer Science & Engineering at Washington University is committed to helping students gain the background they need.

People are attracted to the study of computing for a variety of reasons. Consequently, the department offers a wide variety of academic programs, including a five-course minor, a second major, five undergraduate degrees, combined undergraduate and graduate programs, and several undergraduate research opportunities. Each academic program can be tailored to a student's individual needs.

The breadth of computer science and engineering may be best understood in terms of the general areas of applications, software systems, hardware and theory.

Applications are the ways in which computer technology is applied to solve problems, often in other disciplines. Most applications courses provide background not only in the applications themselves but also in how the applications are designed and implemented. Examples of application areas include artificial intelligence, computer graphics, game design and computational biology.

Software systems are collections of interacting software components that work together to support the needs of computer applications. Courses in this area help students gain a solid understanding of how software systems are designed and implemented. During the process, students develop their own software systems. Examples include operating systems, which manage computational resources; network protocols, which are responsible for the delivery of information; programming languages, which support the construction of software systems and applications; and compilers, which translate computer programs into executable form.

Hardware is the term used to describe the physical and mechanical components of a computer system. Courses in this area provide background in logic circuits, which carry out basic computations; computer architecture, which defines the organization of functional components in a computer system; and peripheral devices such as disks, robot arms that are controlled by the computer system, and sensor systems that gather the information that computer systems use to interact with the physical world.

Theory is the study of the fundamental capabilities and limitations of computer systems. A knowledge of theory helps students choose among competing design alternatives on the basis of their relative efficiency and helps them to verify that their implementations are correct. Theory courses provide background in algorithms, which describe how a computation is to be carried out; data structures, which specify how information is to be organized within the computer; analytical techniques to characterize the time or space requirements of an algorithm or data structure; and verification techniques to prove that solutions are correct.

A well-rounded study of computing includes training in each of these areas. However, depending on a student's educational goals, the student may prefer to concentrate on certain areas for greater depth of knowledge. To help students balance their elective courses, most upper-level departmental courses are classified into one of the following categories: S for software systems, M for machines (hardware), T for theory, or A for applications. If a student's interests are concentrated in the first two areas, a computer engineering degree might be best. Students are encouraged to meet with a faculty adviser in the Department of Computer Science & Engineering to discuss their options and develop a plan consistent with their goals.

Undergraduate Programs

The Department of Computer Science & Engineering (CSE) offers an array of courses that can be taken as requirements or electives for any of the undergraduate degree programs. We offer a Bachelor of Science in Computer Science (BSCS), a Bachelor of Science in Computer Engineering (BSCoE), a Bachelor of Science in Business and Computer Science (CS+Business), a Bachelor of Science in Computer Science + Mathematics (CS+Math), a Bachelor of Science in Computer Science + Economics (CS+Econ), and a Second Major in Computer Science. In addition to these six programs, CSE offers a pre-medical option and combined undergraduate/graduate programs. As a part of our program, each student is assigned an adviser who can help to design an individualized program, monitor a student's progress, and consult about curriculum and career options. Additional information can be found on our CSE website (http://cse.wustl.edu/), or any of the CSE faculty (https://cse.wustl.edu/faculty/Pages/default.aspx) can offer further guidance and information about our programs.

BSCS: The computer science (https://cse.wustl.edu/undergraduate/programs/Pages/bs-in-computer-science.aspx) major is designed for students planning a career in computing.
BScOE: The computer engineering (https://cse.wustl.edu/undergraduate/programs/Pages/bs-in-computer-engineering.aspx) major encompasses studies of hardware, software and systems issues that arise in the design, development and application of computer systems.

CS+Business: This joint major (https://cse.wustl.edu/undergraduate/programs/Pages/bs-business-computer-science.aspx) provides students with the fundamental knowledge and perspectives of computer science and business and of the unique opportunities created by combining them.

CS+Econ: This applied science major (http://bulletin.wustl.edu/undergrad/engineering/computerscience/bs-applied-computer-science-economics/) allows students interested in both economics and computer science to combine these two complementary disciplines efficiently.

CS+Math: This applied science major (https://cse.wustl.edu/undergraduate/programs/Pages/bs-in-computer-science-math.aspx) efficiently captures the intersection of the complementary studies of computer science and math.

Second Major in Computer Science: The second major (https://cse.wustl.edu/undergraduate/programs/Pages/second-major.aspx) provides an opportunity to combine computer science with another degree program. A second major in computer science can expand a student’s career options and enable interdisciplinary study in areas such as cognitive science, computational biology, chemistry, physics, philosophy and linguistics. The second major is also well suited for students planning careers in medicine, law, business, architecture and fine arts.

Pre-Medical Option within Computer Science: Students may pursue a pre-medicine curriculum in conjunction with either the BS degree or the second major in computer science programs. Students interested in the pre-medical option should refer to the McKelvey School of Engineering (http://bulletin.wustl.edu/undergrad/engineering/#premedicaleducation) Bulletin page for details.

Combined Undergraduate and Graduate Study

The Department of Computer Science & Engineering offers in-depth graduate study in many areas. Students entering the graduate programs require a background in computer science fundamentals. Washington University undergraduates seeking admission to the graduate degree program to obtain a master’s degree in computer science or computer engineering do not need to take the Graduate Record Examination (GRE). For more information, contact the department office by email at admissions@cse.wustl.edu or by phone at 314-935-6132.

The Joint Bachelor's/Master's Program

This five-year program that leads to both the bachelor's and master's degrees offers the student an excellent opportunity to combine undergraduate and graduate studies in an integrated curriculum. The combination of the two programs extends the flexibility of the undergraduate curriculum to more advanced studies, thereby enabling students to plan their entire spectrum of computing studies in a more comprehensive educational framework. Consistent with the general requirements (http://bulletin.wustl.edu/undergrad/engineering/requirements/) defined by the McKelvey School of Engineering, a minimum of 150 units is required for completion of the bachelor's/master's program. Provided that the 150-unit requirement is satisfied, up to 6 units of course work acceptable for the master's degree can be counted toward both the bachelor's and master's requirements. Students in the bachelor's/master's program can take advantage of the program's flexibility by taking graduate courses toward the graduate degree while still completing the undergraduate degree requirements.

The bachelor's/master's program offers early admission to the graduate programs in computer science and computer engineering and allows a student to complete the master's degree, typically in only one additional year of study (instead of the usual three semesters). Undergraduate financial support is not extended for the additional semesters to complete the master's degree requirements; however, scholarship support based on the student's cumulative grade-point average, calculated at the end of the junior year, will be awarded automatically during the student's final year of study. Students are classified as graduate students during their final year of study, and their tuition charges are at the graduate student rate. For information about scholarship amounts, please visit the Bachelor's/Master's Program in Engineering webpage (http://engineering.wustl.edu/current-students/student-services/Pages/BS-MS-program.aspx).

If students plan to apply to this program, it is recommended that they complete at least an undergraduate minor in computer science, three additional computer science courses at the 400 level, and one additional course at the 500 level during their first four years. Fall applications for this program open on June 1. Spring applications open on March 1. Students are encouraged to apply to this program by October 1 of the first semester of their senior year, and a minimum GPA of 3.0 is required of all applicants. Bachelor's/master's applications will be accepted until the last day of classes of the semester prior to the student beginning the graduate program.

BS/MBA Program

The growing importance of computer-based information systems in the business environment has produced a sustained high demand for graduates with master's degrees in business administration and undergraduate majors in computer science and engineering.
Students should apply to this joint program by February 1 of their junior year. The application for admission to Olin Business School (http://bulletin.wustl.edu/undergrad/business/) is available through the business school. Applicants are judged on undergraduate performance, GMAT scores, summer and/or co-op work experience, recommendations and a personal interview.

**Undergraduate Courses**

Course requirements for the minor and majors may be fulfilled by CSE 131 Introduction to Computer Science, CSE 132 Introduction to Computer Engineering, CSE 240 Logic and Discrete Mathematics, CSE 247 Data Structures and Algorithms, CSE 347 Analysis of Algorithms, and CSE courses with a letter suffix in any of the following categories: software systems (S), hardware (M), theory (T) and applications (A). In addition, with approval of the instructor, up to 6 units of CSE 400E Independent Study can be used toward the CSE electives of any CSE degree. Other CSE courses provide credit toward graduation but not toward the CSE elective requirements for the second major or the BSCS, BSCoE, CS-Math or CS +Business degrees. Undergraduates are encouraged to consider 500-level courses. If a student is interested in taking a course but is not sure if they have the needed prerequisites, the student should contact the instructor.

**Broadening Experiences**

If a student wants to become involved in computer science or computer engineering research or to gain experience in industry while they are an undergraduate, there are many opportunities to do so. A few of these are listed below.

**Co-op:** The Cooperative Education Program allows a student to get valuable experience working in industry while an undergraduate. A co-op experience can give students another perspective on their education and may lead to full-time employment. Numerous companies participate in this program. More information is available from the Engineering Co-op and Internship Program (https://careercenter.wustl.edu/items/engineering-co-op-program/) that is part of the Career Center in the Danforth University Center, Suite 110.

**Research:** Participating in undergraduate research (https://cse.wustl.edu/undergraduate/Pages/undergraduate-research.aspx) is a great way to learn more about a specific area. Research projects are available either for pay or for credit through CSE 400E Independent Study.

**Study Abroad:** Students in the McKelvey School of Engineering can study abroad in a number of countries and participate in several global experiences to help broaden their educational experience. These opportunities will help students become global citizens who are better able to address current issues.

The study of computer science and engineering is especially well suited and popular for study abroad. Students from our department routinely study abroad in Europe, the United Kingdom, Australia, Israel and many other places. Our department works closely with students to identify courses suitable for computer science credit.

For more information about these programs, please visit the McKelvey School of Engineering website (https://engineering.wustl.edu/our-school/initiatives/Pages/global.aspx).

**Research Opportunities**

The Department of Computer Science & Engineering actively promotes a culture of strong undergraduate participation in research. Many undergraduates work in research labs with state-of-the-art equipment that provides them the opportunity to take part in computer science and computer engineering research. Sensor networks, high-speed routers, specialized FPGA hardware, wireless devices, RF tags, digital cameras, robots, large displays and multiprocessors are just a few of the hardware devices undergraduates often use in their projects. Opportunities for exploring modern software development techniques and specialized software systems further enrich the range of research options and help undergraduates sharpen their design and programming skills.

**Advanced Placement/Proficiency**

Students receiving a 4 or 5 on the AP Computer Science A exam are awarded credit for CSE 131 Introduction to Computer Science. Any student can take the CSE 131 proficiency exam, and a suitable score will waive CSE 131 as a requirement.

Upon request, the computer science department will evaluate a student for proficiency for any of our introductory courses. If a student is determined to be proficient in a given course, that course will be waived (without awarding credit) in the student's degree requirements, and the student will be offered guidance in selecting a more advanced course. Questions should be directed to the associate chair at associatechair@cse.wustl.edu.

**Faculty**

**Chair**

Roch Guérin (https://engineering.wustl.edu/faculty/Roch-Guerin.html)

Harold B. and Adelaide G. Welge Professor of Computer Science
PhD, California Institute of Technology
Computer networks and communication systems
Professors
Sanjoy Baruah (https://engineering.wustl.edu/faculty/Sanjoy-Baruah.html)
PhD, University of Texas at Austin
Real-time and safety-critical system design, cyber-physical systems, scheduling theory, resource allocation and sharing in distributed computing environments
Aaron Bobick (https://engineering.wustl.edu/faculty/Aaron-Bobick.html)
James M. McKelvey Professor and Dean
PhD, Massachusetts Institute of Technology
Computer vision, graphics, human-robot collaboration
Michael R. Brent (https://engineering.wustl.edu/faculty/Michael-Brent.html)
Henry Edwin Sever Professor of Engineering
PhD, Massachusetts Institute of Technology
Systems biology, computational and experimental genomics, mathematical modeling, algorithms for computational biology, bioinformatics
PhD, Washington University
Computational biology, genomics, algorithms for comparing and annotating large biosequences
Roger D. Chamberlain (https://engineering.wustl.edu/faculty/Roger-Chamberlain.html)
DSc, Washington University
Computer engineering, parallel computation, computer architecture, multiprocessor systems
Yixin Chen (https://engineering.wustl.edu/faculty/Yixin-Chen.html)
PhD, University of Illinois at Urbana-Champaign
Mathematical optimization, artificial intelligence, planning and scheduling, data mining, learning data warehousing, operations research, data security
Patrick Crowley (https://engineering.wustl.edu/faculty/Patrick-Crowley.html)
PhD, University of Washington
Computer and network systems, network security
Ron K. Cytron (https://engineering.wustl.edu/faculty/Ron-Cytron.html)
PhD, University of Illinois at Urbana-Champaign
Programming languages, middleware, real-time systems
Sanmay Das
PhD, Massachusetts Institute of Technology
Design of algorithms for complex environments, computational social science, machine learning
Christopher D. Gill (https://engineering.wustl.edu/faculty/Christopher-Gill.html)
DSc, Washington University
Parallel and distributed real-time embedded systems, cyber-physical systems, concurrency platforms and middleware, formal models and analysis of concurrency and timing
Barbara J. & Jerome R. Cox Jr. Professor of Computer Science
PhD, Harvard University
Network security, blockchains, medical systems security, industrial systems security, wireless networks, unmanned aircraft systems, internet of things, telecommunications networks, traffic management
Tao Ju (https://engineering.wustl.edu/faculty/Tao-Ju.html)
PhD, Rice University
Computer graphics, visualization, mesh processing, medical imaging and modeling
Chenyang Lu (https://engineering.wustl.edu/faculty/Chenyang-Lu.html)
Fulgraf Professor in the Department of Computer Science & Engineering
PhD, University of Virginia
Internet of things, real-time, embedded, and cyber-physical systems, cloud and edge computing, wireless sensor networks
Neal Patwari (https://engineering.wustl.edu/faculty/Neal-Patwari.html)
PhD, University of Michigan
Application of statistical signal processing to wireless networks, and radio frequency signals
Weixiong Zhang
PhD, University of California, Los Angeles
Computational biology, genomics, machine learning and data mining, and combinatorial optimization

Associate Professors
Kunal Agrawal (https://engineering.wustl.edu/faculty/Kunal-Agrawal.html)
PhD, Massachusetts Institute of Technology
Parallel computing, cyber-physical systems & sensing, theoretical computer science
Caitlin Kelleher (https://engineering.wustl.edu/faculty/Caitlin-Kelleher.html)
PhD, Carnegie Mellon University
Human-computer interaction, programming environments, and learning environments
PhD, University of Missouri-Rolla
Ultrasonic imaging, medical instrumentation, computer engineering

Yevgeniy Vorobeychik (https://engineering.wustl.edu/faculty/Yevgeniy-Vorobeychik.html)
PhD, University of Michigan
Artificial intelligence, machine learning, computational economics, security and privacy, multi-agent systems

William Yeoh (https://engineering.wustl.edu/faculty/William-Yeoh.html)
PhD, University of Southern California
Artificial intelligence, multi-agent systems, distributed constraint optimization, planning and scheduling

Assistant Professors
Ayan Chakrabarti (https://engineering.wustl.edu/faculty/Ayan-Chakrabarti.html)
PhD, Harvard University
Computer vision computational photography, machine learning

Roman Garnett (https://engineering.wustl.edu/faculty/Roman-Garnett.html)
PhD, University of Oxford
Active learning (especially with atypical objectives), Bayesian optimization, and Bayesian nonparametric analysis

Chien-Ju Ho (https://engineering.wustl.edu/faculty/Chien-Ju-Ho.html)
PhD, University of California, Los Angeles
Design and analysis of human-in-the-loop systems, with techniques from machine learning, algorithmic economics, and online behavioral social science

Brendan Juba (https://engineering.wustl.edu/faculty/Brendan-Juba.html)
PhD, Massachusetts Institute of Technology
Theoretical approaches to artificial intelligence founded on computational complexity theory and theoretical computer science more broadly construed

Ulugbek Kamilov (https://engineering.wustl.edu/faculty/Ulugbek-Kamilov.html)
PhD, École Polytechnique Fédérale de Lausanne, Switzerland
Computational imaging, image and signal processing, machine learning and optimization

PhD, Massachusetts Institute of Technology
Designing linguistics for parallel programming, developing runtime system support for multithreaded software, and building novel mechanisms in operating systems and hardware to efficiently support parallel abstractions

Alvitta Ottley (https://engineering.wustl.edu/faculty/Alvitta-Ottley.html)
PhD, Tufts University
Designing personalized and adaptive visualization systems, including information visualization, human-computer interaction, visual analytics, individual differences, personality, user modeling and adaptive interfaces

Netanel Raviv (https://engineering.wustl.edu/faculty/Netanel-Raviv.html)
PhD, Technion, Haifa, Israel
Mathematical tools for computation, privacy and machine learning

Ning Zhang (https://engineering.wustl.edu/faculty/Ning-Zhang.html)
PhD, Virginia Polytechnic Institute and State University
System security, software security

Teaching Professor
William Siever (https://engineering.wustl.edu/faculty/Bill-Siever.html)
PhD, Missouri University of Science and Technology
Computer architecture, organization, and embedded systems

Professor of the Practice
Dennis Cosgrove (https://engineering.wustl.edu/faculty/Dennis-Cosgrove.html)
BS, University of Virginia
Programming environments and parallel programming

Lecturers
Hila Ben Abraham, Lecturer
PhD, Washington University in St. Louis
Parallel computing, accelerating streaming applications on GPUs, computer and network security, malware analysis

Steve Cole, Senior Lecturer
PhD, Washington University in St. Louis
Parallel computing, accelerating streaming applications on GPUs

Brian Garnett (https://engineering.wustl.edu/faculty/Brian-Garnett.html), Lecturer
PhD, Rutgers University
Discrete mathematics and probability, generally motivated by theoretical computer science

Marion Neumann (https://engineering.wustl.edu/faculty/Marion-Neumann.html), Senior Lecturer
PhD, University of Bonn, Germany
Machine learning with graphs; solving problems in agriculture and robotics
James Orr (https://engineering.wustl.edu/faculty/James-Orr.html), Lecturer
PhD, Washington University
Real-time systems theory and implementation, cyber-physical systems, and operating systems

Jonathan Shidal (https://engineering.wustl.edu/faculty/Jonathan-Shidal.html), Lecturer
PhD, Washington University
Computer architecture and memory management

Douglas Shook (https://engineering.wustl.edu/faculty/Doug-Shook.html), Senior Lecturer
MS, Washington University
Imaging sensor design, compiler design and optimization

Senior Professors

Jonathan S. Turner
PhD, Northwestern University
Design and analysis of internet routers and switching systems, networking and communications, algorithms

Senior Faculty Emeritus

Jerome R. Cox Jr.
ScD, Massachusetts Institute of Technology
Computer system design, computer networking, biomedical computing

Professors Emeriti

Takayuki D. Kimura
PhD, University of Pennsylvania
Communication and computation, visual programming

Seymour V. Pollack
MS, Brooklyn Polytechnic Institute
Intellectual property, information systems

Majors

Please visit the following pages for information about computer science and engineering majors:

• Bachelor of Science in Computer Engineering (http://bulletin.wustl.edu/undergrad/engineering/computerscience/bc-computer-engineering/)
• Bachelor of Science in Computer Science (http://bulletin.wustl.edu/undergrad/engineering/computerscience/bc-computer-science/)
• Bachelor of Science in Computer Science + Economics (http://bulletin.wustl.edu/undergrad/engineering/computerscience/bc-applied-computer-science-economics/)
• Bachelor of Science in Computer Science + Mathematics (http://bulletin.wustl.edu/undergrad/engineering/computerscience/bc-applied-computer-science-math/)

• Bachelor of Science in Business and Computer Science (http://bulletin.wustl.edu/undergrad/engineering/computerscience/bs-business-computer-science/)
• Second Major in Computer Science (http://bulletin.wustl.edu/undergrad/engineering/computerscience/second-major/)
• Second Major in Computer Science + Mathematics

Minors

Please visit the following pages for information about computer science and engineering minors:

• Minor in Computer Science (http://bulletin.wustl.edu/undergrad/engineering/computerscience/minor-computer-science/)
• Minor in Bioinformatics (http://bulletin.wustl.edu/undergrad/engineering/computerscience/minor-bioinformatics/)
• Minor in Human-Computer Interaction (http://bulletin.wustl.edu/undergrad/engineering/computerscience/minor-human-computer-interaction/)

Courses

Visit online course listings to view semester offerings for E81 CSE (https://courses.wustl.edu/CourseInfo.aspx?sch=E&dept=E81&crslvl=1:5).

Prerequisites are advisory in our course listings, but students are cautioned against taking a course without the necessary background. Note that if one course mentions another as its prerequisite, the prerequisites of the latter course are implied to be prerequisites of the former course as well. Students in doubt of possessing the necessary background for a course should correspond with the course’s instructor.

E81 CSE 131 Introduction to Computer Science

An introduction to software concepts and implementation, emphasizing problem solving through abstraction and decomposition. Introduces processes and algorithms, procedural abstraction, data abstraction, encapsulation and object-oriented programming. Recursion, iteration and simple data structures are covered. Concepts and skills are mastered through programming projects, many of which employ graphics to enhance conceptual understanding. Java, an object-oriented programming language, is the vehicle of exploration. Active-learning sessions are conducted in a studio setting in which students interact with each other and the professor to solve problems collaboratively. Prerequisites: Comfort with algebra and geometry at the high school level is assumed. Patience, good planning and organization promote success. This course assumes no prior experience with programming.

Credit 3 units. BU: SCI EN: TU
E81 CSE 131R Seminar: Computer Science I
A seminar and discussion session that complements the material studied in CSE 131. Provides background and breadth for the disciplines of computer science and computer engineering. Features guest lectures and highly interactive discussions of diverse computer science topics. Highly recommended for majors and for any student seeking a broader view of computer science or computer engineering. Pass/Fail only. Credit 1 unit.

E81 CSE 132 Introduction to Computer Engineering
This course introduces students to fundamental concepts in the basic operation of computers, ranging from desktops and servers to microcontrollers and handheld devices. Subjects include digital and analog input/output, sensing the physical world, information representation, basic computer architecture and machine language, time-critical computation, machine-to-machine communication and protocol design. Students will use both desktop systems and handheld microcontrollers for laboratory experiments. Active-learning sessions are conducted in a studio setting in which students interact with each other and the professor to solve problems collaboratively. Prerequisite: CSE 131. Credit 3 units. BU: SCI EN: TU

E81 CSE 132R Seminar: Computer Science II
This course is a seminar and discussion session that complements the material studied in CSE 132. It provides background and breadth for the disciplines of computer science and computer engineering, and it features guest lectures and highly interactive discussions of diverse computer science topics. Highly recommended for majors and for any student seeking a broader view of computer science or computer engineering. Credit 1 unit.

E81 CSE 204A Front-End Web Development
This course explores elementary principles for designing, creating, and publishing effective websites and web application front-ends. Topics include page layout concepts, design principles, HTML, CSS, JavaScript, front-end frameworks like Angular and React, and other development tools. Students apply the topics by creating a series of websites that are judged based on their design and implementation. Prerequisite: CSE 131 or equivalent experience. Credit 3 units. EN: TU

E81 CSE 217A Introduction to Data Science
This course provides an introduction to data science and machine learning, and it focuses on the practical application of models to real-world supervised and unsupervised learning problems. We will discuss methods for linear regression, classification, and clustering and apply them to perform sentiment analysis, implement a recommendation system, and perform image classification or gesture recognition. One of the main objectives of the course is to become familiar with the data science workflow, from posing a problem to understanding and preparing the data, training and evaluating a model, and then presenting and interpreting the results. We will also touch on concepts such as similarity-based learning, feature engineering, data manipulation, and visualization. The course uses Python, which is currently the most popular programming language for data science. Prerequisites: CSE 131, MATH 233, and CSE 247 (can be taken concurrently) Credit 3 units. EN: TU

E81 CSE 222S Internet of Things
With the advent of the Internet of Things, we can address, control, and interconnect formerly isolated objects to create new and interesting applications. In this course we study fundamental technologies behind Internet-of-Things devices, and appcessories, which include smart watches, health monitors, toys, and appliances. In addition to learning about IoT, students gain hands-on experience developing multi-platform solutions that control and communicate with Things using various mobile device friendly interfaces. Students apply their knowledge and skill to develop a project of their choosing using topics from the course. Prerequisite: CSE 132 Revision: 2020-02-26 Credit 3 units. EN: TU

E81 CSE 231S Introduction to Parallel and Concurrent Programming
This course explores concepts, techniques, and design approaches for parallel and concurrent programming. We study how to write programs that make use of multiple processors for responsiveness and that share resources reliably and fairly. Parallel programming concepts include task-level, functional, and loop-level parallelism. Concurrent programming concepts include threads, synchronization, and locks. We cover how to adapt algorithms to achieve determinism and avoid data races and deadlock. Concepts and skills are acquired through the design and implementation of software projects. Prerequisite: CSE 131. Credit 3 units. EN: TU

E81 CSE 232 Programming Skills Workshop
This course provides an overview of practical implementation skills. Topics include compilation and linking, memory management, pointers and references, using code libraries, testing and debugging. Prerequisite: CSE 247. Credit 1 unit.

E81 CSE 240 Logic and Discrete Mathematics
Introduces elements of logic and discrete mathematics that allow reasoning about computational structures and processes. Generally, the areas of discrete structures, proof techniques, probability and computational models are covered. Topics typically include propositional and predicate logic; sets, relations, functions and graphs; proof by contradiction, induction and recursion; finite state machines and regular languages; and introduction to discrete probability, expected value and variance. Prerequisite: CSE 131. Credit 3 units. EN: TU

E81 CSE 247 Data Structures and Algorithms
This course involves the study of fundamental algorithms, data structures, and their effective use in a variety of applications. It emphasizes the importance of data structure choice and implementation for obtaining the most efficient algorithm for solving a given problem. A key component of this course is worst-case asymptotic analysis, which provides a quick and simple method for determining the scalability and effectiveness of an algorithm. Prerequisite: CSE 131 and fluency with summations, derivatives, and proofs by induction. Credit 3 units. EN: TU
E81 CSE 247R Seminar: Data Structures and Algorithms
The content of this seminar will vary by semester, but it will generally complement the material taught in CSE 247 Data Structures and Algorithms. Corequisite: CSE 247.
Credit 1 unit. EN: TU

E81 CSE 256A Introduction to Human-Centered Design
This course provides an introduction to human-centered design through a series of small user interface development projects covering usability topics such as efficiency vs. learnability, walk up and use systems, the habit loop, and information foraging. Projects will begin with reviewing a relevant model of human behavior. Each project will then provide an opportunity to explore how to apply that model in the design of a new user interface. Prerequisite: CSE 131.
Credit 3 units. EN: TU

E81 CSE 260M Introduction to Digital Logic and Computer Design
Introduction to design methods for digital logic and fundamentals of computer architecture. Boolean algebra and logic minimization techniques; sources of delay in combinational circuits and effect on circuit performance; survey of common combinational circuit components; sequential circuit design and analysis; timing analysis of sequential circuits; use of computer-aided design tools for digital logic design (schematic capture, hardware description languages, simulation); design of simple processors and memory subsystems; program execution in simple processors; basic techniques for enhancing processor performance; configurable logic devices. Prerequisites: CSE 131.
Credit 3 units. EN: TU

E81 CSE 311A Introduction to Intelligent Agents Using Science Fiction
The course uses science-fiction short stories, tv episodes, and movies to motivate and introduce fundamental principles and techniques in intelligent agent systems. Topics covered may include game theory, decision theory, machine learning, distributed algorithms, and ethics. Prerequisite: CSE 247
Credit 3 units. EN: BME T, TU

E81 CSE 316A Social Network Analysis
This course looks at social networks and markets through the eyes of a computer scientist. We will look at questions including, "Why are acquaintances rather than friends more likely to get us job opportunities?" and, "Why do the rich get richer?" We begin by studying graph theory (allowing us to study the structure) and game theory (allowing us to study the interactions) of social networks and market behavior at the introductory level. Among other topics, we will study auctions, epidemics, and the structure of the internet (including web searches). This course examines the intersection of computer science, economics, sociology, and applied mathematics. Prerequisites: CSE 240 and CSE 247.
Credit 3 units. EN: BME T, TU

E81 CSE 330S Rapid Prototype Development and Creative Programming
This course uses web development as a vehicle for developing skills in rapid prototyping. Students acquire the skills to build a Linux web server in Apache, to write a website from scratch in PHP, to run an SQL database, to perform scripting in Python, to employ various web frameworks, and to develop modern web applications in client-side and server-side JavaScript. The course culminates with a creative project in which students are able to synthesize the course material into a project of their own interest. The course implements an interactive studio format; after the formal presentation of a topic, students develop a related project under the supervision of the instructor. Prerequisite: CSE 131.
Credit 3 units. EN: BME T, TU

E81 CSE 332S Object-Oriented Software Development Laboratory
Intensive focus on practical aspects of designing, implementing and debugging software, using object-oriented, procedural, and generic programming techniques. The course emphasizes familiarity and proficiency with a wide range of C++ language features through hands-on practice completing studio exercises and lab assignments, supplemented with readings and summary presentations for each session. Prerequisites: CSE 131 and CSE 247
Credit 3 units. EN: BME T, TU

E81 CSE 341T Parallel and Sequential Algorithms
The course aims to teach how to design, analyze and implement parallel algorithms. The emphasis is on teaching fundamental principles and design techniques that easily transfer over to parallel programming. These techniques include divide and conquer, contraction, the greedy method, etc. Prerequisite: CSE 240, 247, Math 310
Credit 3 units. EN: BME T, TU

E81 CSE 347 Analysis of Algorithms
Introduces techniques for the mathematical analysis of algorithms, including randomized algorithms and non-worst-case analyses such as amortized and competitive analysis. Introduces the standard paradigms of divide-and-conquer, greedy, and dynamic programming algorithms, as well as reductions. Also provides an introduction to the study of intractability and techniques to determine when good algorithms cannot be designed. Prerequisites: CSE 240, 247 (or Math 310).
Credit 3 units. EN: BME T, TU

E81 CSE 347R Analysis of Algorithms Recitation
This course is the recitation component of CSE 347. Registration and attendance for 347R is mandatory for students enrolled in 347.

E81 CSE 361S Introduction to Systems Software
Introduction to the hardware and software foundations of computer processing systems. This course provides a programmer's perspective of how computer systems execute programs and store information. The course material aims to enables students to become more effective programmers, especially in dealing with issues of performance, portability and robustness. It also serves as a foundation for other system courses, such as compilers, networks, and operating systems, where a deeper understanding of systems-level issues is required. Topics covered include: machine-level code and its generation by optimizing compilers, performance evaluation and optimization, computer arithmetic, memory organization and management, and supporting concurrent computation. Prerequisites: CSE 131 and CSE 132
Credit 3 units. EN: BME T, TU
E81 CSE 362M Computer Architecture
This course explores the interaction and design philosophy of hardware and software for digital computer systems. Topics include: processor architecture, instruction set architecture, Assembly Language, memory hierarchy design, I/O considerations, and a comparison of computer architectures. Prerequisite: CSE 260M.
Credit 3 units. EN: BME T, TU

E81 CSE 365S Elements of Computing Systems
All computers are made up of 0s and 1s. However, the conceptual gap between the 0s and 1s and the day-to-day operation of modern computers is enormously wide. This fast-paced course aims to bridge the divide by starting with simple logic gates and building up the levels of abstraction until one can create games like Tetris. There is no specific programming language requirement, but some experience with programming is needed. Prerequisite: CSE 247.
Credit 3 units. EN: BME T, TU

E81 CSE 400 Independent Study
Investigation of a topic in computer science and engineering of mutual interest to the student and a mentor. Interested students are encouraged to approach and engage faculty to develop a topic of interest. A form (https://cse.wustl.edu/undergraduate/PublishingImages/Pages/undergraduate-research/Independent%20Study%20Form%20400.pdf) declaring the agreement must be filed in the departmental office. This course carries university credit, but it does not count toward a CSE major or minor. To arrange for CSE major or minor credit for independent study, a student must enroll in CSE 400E instead of CSE 400. Consult also: CSE 400E.
Credit variable, maximum 6 units.

E81 CSE 400E Independent Study
Investigation of a topic in computer science and engineering of mutual interest to the student and a mentor. Interested students are encouraged to approach and engage faculty to develop a topic of interest. A form declaring the agreement (https://cse.wustl.edu/undergraduate/PublishingImages/Pages/undergraduate-research/Independent%20Study%20Form%20400.pdf) must be filed in the departmental office. Credit earned for CSE 400E can be counted toward a student’s major or minor program, with the consent of the student’s advisor. Consult also: CSE 400.
Credit variable, maximum 6 units.

E81 CSE 411A AI and Society
AI has made increasing inroads in a broad array of applications, many that have socially significant implications. This course will study a number of such applications, focusing on issues such as AI used for social good, fairness and accountability of AI, and potential security implications of AI systems. Prerequisite: CSE 247.
Credit 3 units. EN: TU

E81 CSE 416A Analysis of Network Data
This course examines complex systems through the eyes of a computer scientist. We will use the representative power of graphs to model networks of social, technological, or biological interactions. Network analysis provides many computational, algorithmic, and modeling challenges. We begin by studying graph theory, allowing us to quantify the structure and interactions of social and other networks. We will then explore how to practically analyze network data and how to reason about it through mathematical models of network structure and evolution. We will also investigate algorithms that extract basic properties of networks in order to find communities and infer node properties. Finally, we will study a range of applications including robustness and fragility of networks such as the internet, spreading processes used to study epidemiology or viral marketing, and the ranking of webpages based on the structure of the webgraph. This course combines concepts from computer science and applied mathematics to study networked systems using data mining. Prerequisites: CSE 247, ESE 326, MATH 309, and programming experience. (Note: We will parse data and analyze networks using Python.)
Credit 3 units. EN: BME T, TU

E81 CSE 417T Introduction to Machine Learning
The field of machine learning is concerned with the question of how to construct computer programs that automatically improve with experience. This course is a broad introduction to machine learning, covering the foundations of supervised learning and important unsupervised learning algorithms. Topics to be covered are the theory of generalization (including VC-dimension, the bias-variance tradeoff, validation, and regularization) and linear and non-linear learning models (including linear and logistic regression, decision trees, ensemble methods, neural networks, nearest-neighbor methods, and support vector machines). Prerequisites: CSE 247, ESE 326 (or Math 3200), Math 233, and Math 309 (can be taken concurrently).
Credit 3 units. EN: BME T, TU

E81 CSE 422S Operating Systems Organization
This course involves a hands-on exploration of core OS abstractions, mechanisms and policies in the context of the Linux kernel. Readings, lecture material, studio exercises, and lab assignments are closely integrated in an active-learning environment in which students gain experience and proficiency writing OS code, as well as tracing and evaluating OS operations via user-level programs and kernel-level monitoring tools. Topics include: system calls, interrupt handling, kernel modules, concurrency and synchronization, proportional and priority-based scheduling of processes and threads, I/O facilities, memory management, virtual memory, device management, and file system organization. Prerequisite: CSE 361S.
Credit 3 units. EN: BME T, TU

E81 CSE 425S Programming Systems and Languages
A systematic study of the principles, concepts and mechanisms of computer programming languages; their syntax, semantics and pragmatics; the processing and interpretation of computer programs; programming paradigms; and language design. Illustrative examples are selected from a variety of programming language paradigms. Prerequisites: CSE 332S.
Credit 3 units. EN: BME T, TU

E81 CSE 427S Cloud Computing with Big Data Applications
This course provides a comprehensive introduction to applied parallel computing using the MapReduce programming model facilitating large-scale data management and processing. There will be an emphasis on hands-on experience working with the Hadoop architecture, an open-source software framework written in Java for distributed storage and processing of very large data sets on computer clusters. Further, we will make use of related
big data technologies from the Hadoop ecosystem of tools, such as Hive, Impala, and Pig in developing analytics and solving problems faced by enterprises today. Prerequisites: CSE 131, CSE 247, and CSE 330. Credit 3 units. EN: BME T, TU

E81 CSE 431S Translation of Computer Languages
The theory of language recognition and translation is introduced in support of compiler construction for modern programming languages. Topics include syntactic and semantic analysis, symbol table management, code generation, and runtime libraries. A variety of parsing methods is covered, including top-down and bottom-up. Machine problems culminate in the course project, for which students construct a working compiler. Prerequisites: CSE 240 and CSE 247. Credit 3 units. EN: BME T, TU

E81 CSE 433R Seminar: Capture The Flag (CTF) Studio
This course provides a collaborative studio space for hands-on practice solving security-relevant puzzles in "Capture The Flag" (CTF) format. Students will engage CTF challenges individually and in teams, and online CTF resources requiring (free) account signup may be used. Students will be encouraged to attempt challenges commensurate with their ability, but no prior CTF experience or security knowledge is assumed. The goal of the course is to build skills in the fundamentals of security analysis, including usage of the Linux command line and console-based security tools, creativity in applying theoretical knowledge to practical challenges, and confidence in approaching under-specified problems. Prerequisite/corequisite: CSE 433S or equivalent. Credit 1 unit.

E81 CSE 433S Introduction to Computer Security
Secure computing requires the secure design, implementation, and use of systems and algorithms across many areas of computer science. Fundamentals of secure computing such as trust models and cryptography will lay the groundwork for studying key topics in the security of systems, networking, web design, machine learning algorithms, mobile applications, and physical devices. Human factors, privacy, and the law will also be considered. Hands-on practice exploring vulnerabilities and defenses using Linux, C, and Python in studios and lab assignments is a key component of the course. Prerequisites: CSE 247 and either CSE 361 or CSE 332. Credit 3 units. EN: TU

E81 CSE 434S Reverse Engineering and Malware Analysis
This course will focus on reverse engineering and malware analysis techniques. Reverse engineering -- the process of deconstructing an object to reveal its design and architecture -- is an essential skill in the information security community. Although hackers often use reverse engineering tools to discover and exploit vulnerabilities, security analysts and researchers must use reverse engineering techniques to find what a specific malware does, how it does it, and how it got into the system. In this course, we will explore reverse engineering techniques and tools, focusing on malware analysis. Students will use and write software during in-class studios and homework assignments to illustrate mastery of the material. Prerequisite: CSE 361S. Credit 3 units. EN: TU

E81 CSE 437R Technical Interviewing
Students will study, give, and receive technical interviews in this seminar course. The focus will be on improving student performance in a technical interview setting, with the goal of making our students as comfortable and agile as possible with technical interviews. Professionals from the local and extended Washington University community will mentor the students in this seminar. Credit 2 units.

E81 CSE 437S Software Engineering Workshop
An introduction and exploration of concepts and issues related to large-scale software systems development. Areas of exploration include technical complexities, organization issues, and communication techniques for large-scale development. Students participate through teams emulating industrial development. The projects cover the principal system development life-cycle phases from requirements analysis, to software design, and to final implementation. Issues relating to real-time control systems, human factors, reliability, performance, operating costs, maintainability and others are addressed and resolved in a reasonable manner. Prerequisite: junior or senior standing. Credit 3 units. EN: BME T, TU

E81 CSE 438S Mobile Application Development
Learn how to create iOS apps in the Swift programming language. This course covers a variety of topics in the development of modern mobile applications, with a focus on hands-on projects. Students will create multiple fully-functional apps from scratch. The course emphasizes object-oriented design patterns and real-world development techniques. Time is provided at the end of the course for students to work on a project of their own interest. Prerequisite: CSE 247. Credit 3 units. EN: BME T, TU

E81 CSE 439S Mobile Application Development II
In this course, students will work in groups to design, develop, test, publish, and market an iOS mobile application. Throughout the semester, students will operate in different roles on a team, serving as lead developer, tester, and project manager. Topics include how to publish a mobile application on an app store, APIs and tools for testing and debugging, and popular cloud-based SDKs used by developers. This course requires completion of the iOS version of CSE 438 Mobile Application Development or the appropriate background knowledge of the iOS platform. Credit 3 units. EN: TU

E81 CSE 442T Introduction to Cryptography
This course is an introduction to modern cryptography, with an emphasis on its theoretical foundations. Topics will include one-way functions, pseudorandom generators, public key encryption, digital signatures, and zero-knowledge proofs. Prerequisites: CSE 240 and CSE 247. Credit 3 units. EN: TU

E81 CSE 450A Video Game Programming
This course teaches the core aspects of a video game developer's toolkit. Students work in groups and with a large game software engine to create and playtest a full-featured video game. Students will explore topics around the design
E81 CSE 451A Video Game Programming II
This course is a continuation of CSE 450A Video Game Programming I. Students will work in groups and with a large game software engine to make a full-featured video game. Students will have the opportunity to work on topics in graphics, artificial intelligence, networking, physics, user interface design, and other topics. Prerequisite: CSE 450A and permission of instructor. Credit 3 units. EN: TU

E81 CSE 452A Computer Graphics
Introduction to computer graphics. The course covers fundamental concepts, data structures and algorithms related to the construction, display and manipulation of three-dimensional objects. Topics include scan-convolution, basic image processing, transformations, scene graphs, camera projections, local and global rendering, fractals, and parametric curves and surfaces. Students develop interactive graphics programs using C++ language. Prerequisites: CSE 332S and Math 309. Credit 3 units. EN: TU

E81 CSE 457A Introduction to Visualization
Disciplines such as medicine, business, science, and government are producing enormous amounts of data with increasing volume and complexity. In this course, students will study the principles for transforming abstract data into useful information visualizations. They will learn about the state of the art in visualization research and development and gain hands-on experience with designing and developing interactive visualization tools for the web. They will also learn how to critique existing visualizations and how to evaluate the systems they build. Topics include design, data mapping, visual perception, and interaction. Prerequisite: CSE 330S. Credit 3 units. EN: TU

E81 CSE 460T Switching Theory
Advanced topics in switching theory as employed in the synthesis, analysis and design of information processing systems. Combinational techniques: minimization, multiple output networks, state identification and fault detection, hazards, testability and design for test are examined. Sequential techniques: synchronous circuits, machine minimization, optimal state assignment, asynchronous circuits, and built-in self-test techniques. Prerequisite: CSE 260M. Credit 3 units. EN: TU

E81 CSE 462M Computer Systems Design
Introduction to modern design practices, including FPGA and PCB design methodologies. Student teams use Xilinx Vivado for HDL-based FPGA design and simulation; they also perform schematic capture, PCB layout, fabrication, and testing of the hardware portion of a selected computation system. The software portion of the project uses Microsoft Visual Studio to develop a user interface and any additional support software required to demonstrate final projects to the faculty during finals week. Prerequisite: CSE 361S and 362M from Washington University in St. Louis or permission of the instructor. Credit 3 units. EN: TU

E81 CSE 465M Digital Systems Laboratory
Hardware/software co-design; processor interfacing; procedures for reliable digital design, both combinational and sequential; understanding manufacturers' specifications; use of test equipment. Several single-period laboratory exercises, several design projects, and application of microprocessors in digital design. One lecture and one laboratory period a week. Prerequisites: ESE 260. Same as E35 ESE 465 Credit 3 units. EN: BME T, TU

E81 CSE 467S Embedded Computing Systems
This course introduces the issues, challenges, and methods for designing embedded computing systems -- systems designed to serve a particular application and which incorporate the use of digital processing devices. Examples of embedded systems include PDAs, cellular phones, appliances, game consoles, automobiles, and iPods. Emphasis is given to aspects of design that are distinct to embedded systems. The course examines hardware, software, and system-level design. Hardware topics include microcontrollers, digital signal processors, memory hierarchy, and I/O. Software issues include languages, run-time environments, and program analysis. System-level topics include real-time operating systems, scheduling, power management, and wireless sensor networks. Students will perform a course project on a real wireless sensor network testbed. Prerequisite: CSE 361S. Credit 3 units. EN: TU

E81 CSE 468T Introduction to Quantum Computing
This course introduces students to quantum computing, which leverages the effects of quantum-mechanical phenomena to solve problems. The course begins with material from physics that demonstrates the presence of quantum effects. Mathematical abstractions of quantum gates are studied with the goal of developing the skills needed to reason about existing quantum circuits and to develop new quantum circuits as required to solve problems. Students will develop a quantum-computer simulator and make use of open simulators as well as actual devices that can realize quantum circuits on the internet. Prerequisites: CSE247, Math 309, and either Math 3200 or ESE 326. Credit 3 units. EN: TU

E81 CSE 469S Security of the Internet of Things and Embedded System Security
This course presents a deep dive into the emerging world of the "Internet of things" from a cybersecurity perspective. With billions of internet-enabled devices projected to impact every nook and cranny of modern existence, the concomitant security challenge portends to become dazzlingly complex. Students will learn the fundamentals of internet of things architecture and operations from a layered perspective and focus on identifying, assessing, and mitigating the threats and vulnerabilities therein. Through a blend of lecture and hands-on studios, students will gain
proficiency in the range of approaches, methods, and techniques required to address embedded systems security and secure the internet of things using actual devices from both hardware and software perspectives and across a range of applications.

Credit 3 units. EN: TU

E81 CSE 473S Introduction to Computer Networks
A broad overview of computer networking. Topics include history, protocols, Hyper Text Transfer Protocol (HTTP), File Transfer Protocol (FTP), Simple Mail Transfer Protocol (SMTP), Domain Name System (DNS), peer-to-peer (P2P), transport layer design issues, transport layer protocols, Transmission Control Protocol (TCP), User Datagram Protocol (UDP), TCP congestion control, network layer, Internet Protocol version 4 (IPv4), Internet Control Message Protocol (ICMP), Internet Protocol version 6 (IPv6), routing algorithms, routing protocols, Open Shortest Path First (OSPF), Routing Information Protocol (RIP), Border Gateway Protocol (BGP), datalink layer and local area networks carrier sense multiple access with collision detection (CSMA/CD), Ethernet, virtual local area networks (VLANs), Point-to-Point Protocol (PPP), Multi-Protocol Label Switching, wireless and mobile networks, multimedia networking, security in computer networks, cryptography, and network management. Prerequisite: CSE 247; CSE 132 is suggested but not required.

Credit 3 units. EN: BME T, TU

E81 CSE 497 Senior Project I
Implementation of a substantive project on an individual basis, involving one or more major areas in computer science. Problems pursued under this framework may be predominantly analytical, involving the exploration and extension of theoretical structures, or they may pivot around the design/development of solutions for particular applications drawn from areas throughout the University and/or the community. In either case, the project serves as a focal point for crystallizing the concepts, techniques, and methodologies encountered throughout the curriculum. Students intending to take CSE 497-498 must submit a project proposal form (PDF) (https://cse.wustl.edu/undergraduate/PublishingImages/Pages/undergraduate-research/Independent%20Study%20Form%20400.pdf) for approval by the department during the spring semester of the junior year. Prerequisite: senior standing.

Credit 3 units.

E81 CSE 498 Senior Project II
Implementation of a substantive project on an individual basis, involving one or more major areas in computer science. Problems pursued under this framework may be predominantly analytical, involving the exploration and extension of theoretical structures, or they may pivot around the design/development of solutions for particular applications drawn from areas throughout the University and/or the community. In either case, the project serves as a focal point for crystallizing the concepts, techniques, and methodologies encountered throughout the curriculum. Students intending to take CSE 497-498 must submit a project proposal form (PDF) (https://cse.wustl.edu/undergraduate/PublishingImages/Pages/undergraduate-research/Independent%20Study%20Form%20400.pdf) for approval by the department during the spring semester of the junior year. Prerequisite: senior standing.

Credit 3 units.

E81 CSE 499 Undergraduate Honors Thesis
Working closely with a faculty member, the student investigates an original idea (algorithm, model technique, etc.), including a study of its possible implications, its potential application and its relationship to previous related work reported in the literature. Contributions and results from this investigation are synthesized and compiled into a publication-quality research paper presenting the new idea. Prerequisites: a strong academic record and permission of instructor.

Credit 3 units.

E81 CSE 500 Independent Study
Proposal form can be located at https://cse.wustl.edu/undergraduate/PublishingImages/Pages/undergraduate-research/Independent%20Study%20Form%20400.pdf
Credit variable, maximum 3 units.

E81 CSE 501N Introduction to Computer Science
An introduction to software concepts and implementation, emphasizing problem solving through abstraction and decomposition. Introduces processes and algorithms, procedural abstraction, data abstraction, encapsulation, and object-oriented programming. Recursion, iteration, and simple data structures are covered. Concepts and skills are mastered through programming projects, many of which employ graphics to enhance conceptual understanding. Java, an object-oriented programming language, is the vehicle of exploration. Active-learning sessions are conducted in a studio setting in which students interact with each other and the professor to solve problems collaboratively. Prerequisites: Comfort with algebra and geometry at the high school level is assumed. Patience, good planning, and organization will promote success. This course assumes no prior experience with programming.

Same as E81 CSE 131
Credit 3 units. BU: SCI EN: TU

E81 CSE 502N Data Structures and Algorithms
This course involves the study of fundamental algorithms, data structures, and their effective use in a variety of applications. It emphasizes the importance of data structure choice and implementation for obtaining the most efficient algorithm for solving a given problem. A key component of this course is worst-case asymptotic analysis, which provides a quick and simple method for determining the scalability and effectiveness of an algorithm. Prerequisite: CSE 131 and fluency with summations, derivatives, and proofs by induction.

Same as E81 CSE 247
Credit 3 units. EN: TU

E81 CSE 503S Rapid Prototype Development and Creative Programming
This course uses web development as a vehicle for developing skills in rapid prototyping. Students acquire the skills to build a Linux web server in Apache, to write a website from scratch in PHP, to run an SQL database, to perform scripting in Python, to employ various web frameworks, and to develop modern web applications in client-side and server-side JavaScript. The course culminates with a creative project in which students are able to synthesize the course material into a project of their own interest. The course implements an interactive studio format: after the formal presentation of a topic, students develop a related project under the supervision of the instructor. Prerequisite: CSE 131.

Same as E81 CSE 330S
E81 CSE 504N Object-Oriented Software Development Laboratory
Intensive focus on practical aspects of designing, implementing and debugging software, using object-oriented, procedural, and generic programming techniques. The course emphasizes familiarity and proficiency with a wide range of C++ language features through hands-on practice completing studio exercises and lab assignments, supplemented with readings and summary presentations for each session. Prerequisites: CSE 131 and CSE 247
Same as E81 CSE 332S
Credit 3 units. EN: BME T, TU

E81 CSE 505N Introduction to Digital Logic and Computer Design
Introduction to design methods for digital logic and fundamentals of computer architecture. Boolean algebra and logic minimization techniques; sources of delay in combinational circuits and effect on circuit performance; survey of common combinational circuit components; sequential circuit design and analysis; timing analysis of sequential circuits; use of computer-aided design tools for digital logic design (schematic capture, hardware description languages, simulation); design of simple processors and memory subsystems; program execution in simple processors; basic techniques for enhancing processor performance; configurable logic devices. Prerequisite: CSE 131.
Same as E81 CSE 260M
Credit 3 units. EN: BME T, TU

E81 CSE 507A Technology Entrepreneurship
This is a course for students who plan to be or to work with entrepreneurs. An entrepreneurial mindset is needed to create or grow economically viable enterprises, be they new companies, new groups within companies, or new university laboratories. This course aims to cultivate an entrepreneurial perspective, with particular emphasis on information technology (IT)-related activities. The course is jointly offered for business and CSE students, allowing for acculturation between these disciplines. In addition to an introductory treatment of business and technology fundamentals, course topics will include business ethics, opportunity assessment, team formation, financing, intellectual property, and technology transfer. The course will feature significant participant and guest instruction from experienced practitioners.
Credit 3 units.

E81 CSE 511A Introduction to Artificial Intelligence
The discipline of artificial intelligence (AI) is concerned with building systems that think and act like humans or rationally on some absolute scale. This course is an introduction to the field, with special emphasis on sound modern methods. The topics include knowledge representation, problem solving via search, game playing, logical and probabilistic reasoning, planning, dynamic programming, and reinforcement learning. Programming exercises concretize the key methods. The course targets graduate students and advanced undergraduates. Evaluation is based on written and programming assignments, a midterm exam and a final exam. Prerequisites: CSE 247, ESE 326, Math 233.
Credit 3 units. EN: TU

E81 CSE 513T Theory of Artificial Intelligence and Machine Learning
Mathematical foundations for Artificial Intelligence and Machine Learning. An introduction to the PAC-Semantics ("Probably Approximately Correct") as a common semantics for knowledge obtained from learning and declarative sources, and the computational problems underlying the acquisition and processing of such knowledge. We emphasize the design and analysis of efficient algorithms for these problems, and examine for which representations these problems are known or believed to be tractable. Prerequisite: CSE 347.
Credit 3 units. EN: BME T, TU

E81 CSE 514A Data Mining
With the vast advancement in science and technology, data acquisition in large quantities are routinely done in many fields. Examples of large data include various types of data on the internet, high-throughput sequencing data in biology and medicine, extraterrestrial data from telescopes in astronomy, and images from surveillance camera in security. Mining a large amount of data through data mining has become an effective means to extracting knowledge from data. This course introduces the basic concepts and methods for data mining and provides hands-on experience for processing, analyzing and modeling structured and unstructured data. Homework problems, exams and programming assignments will be administrated throughout the course to enhance the learning. Prerequisites: CSE 247, ESE 326 (or Math 3200), Math 233
Credit 3 units. EN: BME T, TU

E81 CSE 515T Bayesian Methods in Machine Learning
This course will cover machine learning from a Bayesian probabilistic perspective. Bayesian probability allows us to model and reason about all types of uncertainty. The result is a powerful, consistent framework for approaching many problems that arise in machine learning, including parameter estimation, model comparison, and decision making. We will begin with a high-level introduction to Bayesian inference, then proceed to cover more-advanced topics. These will include inference techniques (exact, MAP, sampling methods, the Laplace approximation, etc.), Bayesian decision theory, Bayesian model comparison, Bayesian nonparametrics, and Bayesian optimization. Prerequisites: CSE 417T, ESE 326
Credit 3 units. EN: BME T, TU

E81 CSE 516T Multi-Agent Systems
This course introduces the fundamental techniques and concepts needed to study multi-agent systems, in which multiple autonomous entities with different information sets and goals interact. We will study algorithmic, mathematical, and game-theoretic foundations, and how these foundations can help us understand and design systems ranging from robot teams to online markets to social computing platforms. Topics covered may include game theory, distributed optimization, multi-agent learning and decision-making, preference elicitation and aggregation, mechanism design, and incentives in social computing systems. Prerequisites: CSE 347 (may be taken concurrently), ESE 326 (or Math 3200), and Math 233 or equivalents. Some prior exposure to artificial intelligence, machine learning, game theory, and microeconomics may be helpful, but is not required.
Credit 3 units. EN: BME T, TU
E81 CSE 517A Machine Learning
This course assumes a basic understanding of machine learning and covers advanced topics at the frontier of the field in-depth. Topics to be covered include kernel methods, support vector machines, Bayesian networks, convolutional neural networks, recurrent neural networks, deep learning, and unsupervised learning. Depending on developments in the field, the course will also cover some advanced topics, which may include learning from structured data, active learning, and practical machine learning (feature selection, dimensionality reduction). Prerequisites: CSE 417T.
Credit 3 units. EN: TU

E81 CSE 518A Human-in-the-Loop Computation
This course is an exploration of the opportunities and challenges of human-in-the-loop computation, an emerging field that examines how humans and computers can work together to solve problems neither can yet solve alone. We will explore ways in which techniques from machine learning, game theory, optimization, online behavioral social science, and human-computer interactions can be used to model and analyze human-in-the-loop systems such as crowdsourcing markets, prediction markets, and user-generated content platforms. We will also look into recent developments in the interactions between humans and AIs, such as learning with the presence of strategic behavior and ethical issues in AI systems. Prerequisites: CSE 247, ESE 326, Math 233, and Math 309.
Credit 3 units. EN: TU

E81 CSE 519T Advanced Machine Learning
This course provides a close look at advanced machine learning algorithms -- their theoretical guarantees (computational learning theory) and tricks to make them work in practice. In addition, this course focuses on more specialized learning settings, including unsupervised learning, semi-supervised learning, and domain adaptation, multi-task learning, structured prediction, metric learning and learning of data representations. Learning approaches may include graphical models, non-parametric Bayesian statistics, and technical topics such as sampling, approximate inference and non-linear function optimization. Mathematical maturity and general familiarity of machine learning is required. Prerequisites: CSE 511A, CSE 517A, and CSE 571A
Credit 3 units. EN: TU

E81 CSE 520S Real-Time Systems
This course covers software systems and network technologies for real-time applications such as automobiles, avionics, industrial automation and Internet of Things. Topics include real-time scheduling, real-time operating systems and middleware, Quality of Service, industrial networks and real-time cloud computing. Prerequisite: CSE 361S
Credit 3 units. EN: BME T, TU

E81 CSE 521S Wireless Sensor Networks
Dense collections of smart sensors networked to form self-configuring pervasive computing systems provide a basis for a new computing paradigm that challenges many classical approaches to distributed computing. Naming, wireless networking protocols, data management, and approaches to dependability, real-time, security, and middleware services all fundamentally change when confronted with this new environment. Embedded sensor networks and pervasive computing are among the most exciting research areas with many open research questions. This class will study a large number of research papers that deal with various aspects of wireless sensor networks. Students will perform a project on a real wireless sensor network comprised of tiny devices each consisted of sensors, a radio transceiver, and a microcontroller. Prerequisites: CSE 361S.
Credit 3 units. EN: BME T, TU

E81 CSE 522S Advanced Operating Systems
This course offers an in-depth hands-on exploration of core OS abstractions, mechanisms and policies, with an increasing focus on understanding and evaluating their behaviors and interactions. Readings, lecture material, studio exercises, and lab assignments are closely integrated in an active-learning environment in which students gain experience and proficiency writing, tracing, and evaluating user-space and kernel-space code. Topics include: inter-process communication, real-time systems, memory forensics, file-system forensics, timing forensics, process and thread forensics, hypervisor forensics, and managing internal or external causes of anomalous behavior. Prerequisite: CSE 422S.
Credit 3 units. EN: BME T, TU

E81 CSE 523S Systems Security
This course examines the intersection between computer design and information security. While performance and efficiency in digital systems have improved markedly in recent decades, computer security has worsened overall in this time frame. To understand why, we will explore the role that design choices play in the security characteristics of modern computer and network systems. Students will use and write software to illustrate mastery of the material. Projects will include identifying security vulnerabilities, exploiting vulnerabilities, and detecting and defending against exploits. Prerequisite: CSE 361S.
Credit 3 units. EN: BME T, TU

E81 CSE 530S Database Management Systems
A study of data models and the database management systems that support these data models. The design theory for databases is developed and various tools are utilized to apply the theory. General query languages are studied and techniques for query optimization are investigated. Integrity and security requirements are studied in the context of concurrent operations on a database, where the database may be distributed over one or more locations. The unique requirements for engineering design databases, image databases, and long transaction systems are analyzed. Prerequisite: CSE 247.
Credit 3 units.

E81 CSE 532S Advanced Multiparadigm Software Development
Intensive focus on advanced design and implementation of concurrent and distributed system software in C++. Topics covered include concurrency and synchronization features and software architecture patterns. Prerequisites: CSE 332S or graduate standing and strong familiarity with C++; and CSE 422S.
Credit 3 units. EN: BME T, TU

E81 CSE 534A Large-Scale Optimization for Data Science
Large-scale optimization is an essential component of modern data science, artificial intelligence, and machine learning. This graduate-level course rigorously introduces optimization methods that are suitable for large-scale problems arising in...
these areas. Students will learn several algorithms suitable for both smooth and nonsmooth optimization, including gradient methods, proximal methods, mirror descent, Nesterov’s acceleration, ADMM, quasi-Newton methods, stochastic optimization, variance reduction, and distributed optimization. Throughout the class, we will discuss the efficacy of these methods in concrete data science problems, under appropriate statistical models. Students will be required to program in Python or MATLAB. Prerequisites: CSE 247, Math 309, Math 3200 or ESE 326.
Credit 3 units. EN: TU

E81 CSE 538T Advanced Secure Software Engineering
The aim of this course is to provide students with broader and deeper knowledge as well as hands-on experience in understanding security techniques and methods needed in software development. Students complete an independent research project which will involve synthesizing multiple software security techniques and applying them to an actual software program or system.
Credit 3 units. EN: TU

E81 CSE 539S Concepts in Multicore Computing
Nowadays, the vast majority of computer systems are built using multicore processor chips. This fundamental shift in hardware design impacts all areas of computer science - one must write parallel programs in order to unlock the computational power provided by modern hardware. The goal of this course is to study concepts in multicore computing. We will examine the implications of the multicore hardware design, discuss challenges in writing high performance software, and study emerging technologies relevant to developing software for multicore systems. Topics include memory hierarchy, cache coherence protocol, memory models, scheduling, high-level parallel language models, concurrent programming (synchronization and concurrent data structures), algorithms for debugging parallel software, and performance analysis.
Prerequisites: CSE 247 and CSE 361S.
Credit 3 units. EN: BME T, TU

E81 CSE 540T Introduction to Formal Languages and Automata
Provides a broad coverage of fundamental algorithm design techniques, with a focus on developing efficient algorithms for solving combinatorial and optimization problems. The topics covered include the review of greedy algorithms, dynamic programming, NP-completeness, approximation algorithms, the use of linear and convex programming for approximation, and online algorithms. Throughout the course, there is an emphasis on correctness proofs and the ability to apply the techniques taught to design efficient algorithms for problems from a wide variety of application areas. Prerequisite: CSE 347.
Credit 3 units. EN: BME T, TU

E81 CSE 541T Advanced Algorithms
Modern computing systems consist of multiple interconnected components that all influence performance. The focus of this course is on developing modeling tools aimed at understanding how to design and provision such systems to meet certain performance or efficiency targets and the trade-offs involved.
Prerequisite: CSE 240 and CSE 247.
Credit 3 units. EN: TU

E81 CSE 543T Algorithms for Nonlinear Optimization
The course will provide an in-depth coverage of modern algorithms for the numerical solution of multidimensional optimization problems. Unconstrained optimization techniques including Gradient methods, Newton’s methods, Quasi-Newton methods, and conjugate methods will be introduced. The emphasis is on constrained optimization techniques: Lagrange theory, Lagrangian methods, penalty methods, sequential quadratic programming, primal-dual methods, duality theory, nondifferentiable dual methods, and decomposition methods.
The course will also discuss applications in engineering systems and use of state-of-the-art computer codes. Special topics may include large-scale systems, parallel optimization, and convex optimization.
Prerequisites: Calculus I and Math 309.
Credit 3 units.
E81 CSE 549T Theory of Parallel Systems
The course covers various aspects of parallel processing such as algorithms, schedulers and systems from a theoretical perspective. We will cover both classic and recent results in parallel computing. Topics include parallel algorithms and analysis in the work/span model, scheduling algorithms, external memory algorithms and their analysis, cache-coherence protocols, etc. The focus will be on design and analysis. Prerequisite: CSE 247. Credit 3 units. EN: BME T, TU

E81 CSE 554A Geometric Computing for Biomedicine
With the advance of imaging technologies deployed in medicine, engineering and science, there is a rapidly increasing amount of spatial data sets (images, volumes, point clouds, etc.) that need to be processed, visualized, and analyzed. This course will focus on a number of geometry-related computing problems that are essential in the knowledge discovery process in various spatial-data-driven biomedical applications. These problems include visualization, segmentation, mesh construction and processing, shape representation and analysis. The course consists of lectures that cover theories and algorithms, and a series of hands-on programming projects using real-world data collected by various imaging techniques (CT, MRI, electron cryo-microscopy, etc.). Prerequisite: CSE332 (or proficiency in programming in C++ or Java or Python) and CSE247. Credit 3 units. EN: BME T, TU

E81 CSE 555A Computational Photography
Computational Photography describes the convergence of computer graphics, computer vision, and the internet with photography. Its goal is to overcome the limitations of traditional photography using computational techniques to enhance the way we capture, manipulate and interact with visual media. In this course, we study many interesting, recent image-based algorithms and implement them to the degree that is possible. Topics may include: cameras and image formation, human visual perception, image processing (filtering, pyramids), image blending and compositing, image retargeting, texture synthesis and transfer, image completion/inpainting, super-resolution, deblurring, denoising, image-based lighting and rendering, high dynamic range, depth and defocus, flash/no flash photography, coded aperture photography, single/multiview reconstruction, photo quality assessment, non photorealistic rendering, modeling and synthesis using internet data, and others. Prerequisites: CSE 452A, CSE 554A, or CSE 559A. Credit 3 units. EN: BME T, TU

E81 CSE 556A Human-Computer Interaction Methods
An introduction to user centered design processes. The course covers a variety of HCI techniques for use at different stages in the software development cycle, including techniques that can be used with and without users. Students will gain experience using these techniques through in-class exercises and then apply them in greater depth through a semester long interface development project. Students who enroll in this course are expected to be comfortable with building user interfaces in at least one framework and willing to learn whatever framework is most appropriate for their project. Over the course of the semester, students will be expected to present their interface evaluation results through written reports and in class presentations. Prerequisites: 3xxS or 4xxS. Credit 3 units. EN: BME T, TU

E81 CSE 557A Advanced Visualization
We are in an era where it is possible to have all of the world's information at our fingertips. However, the more information we can access, the more difficult it is to obtain a holistic view of the data or to determine what's important to make decisions. Computer-based visualization systems provide the opportunity to represent large or complex data visually to aid comprehension and cognition. In this course, we learn about the state-of-the-art in visualization research and gain hands-on experience with the research pipeline. We also learn how to critique existing work and how to formulate and explore sound research questions. We will cover advanced Visualization topics including User Modeling, Adaptation, Personalization, Perception, and Visual Analytics for Non-Experts. Prerequisites: CSE 457A or permission of instructor. Credit 3 units. EN: BME T, TU

E81 CSE 559A Computer Vision
This course introduces the fundamentals of designing computer vision systems; that can "look at" images and videos and reason about the physical objects and scenes they represent. Topics include image restoration and enhancement; estimation of color, shape, geometry, and motion from images; and image segmentation, recognition, and classification. The focus of the course will be on the mathematical tools and intuition underlying algorithms for these tasks: models for the physics and geometry of image formation, and statistical and machine learning-based techniques for inference. Prerequisites: Math 309 or ESE 318 or equivalent; Math 3200 or ESE 326 or equivalent; and CSE 247 or equivalent. Credit 3 units. EN: BME T, TU

E81 CSE 560M Computer Systems Architecture I
An exploration of the central issues in computer architecture: instruction set design, addressing and register set design, control unit design, microprogramming, memory hierarchies (cache and main memories, mass storage, virtual memory), pipelining, and bus organization. The course emphasizes understanding the performance implications of design choices, using architecture modeling and evaluation using VHDL and/or instruction set simulation. Prerequisites: CSE 361S and CSE 260M. Credit 3 units. EN: BME T, TU

E81 CSE 566S High Performance Computer Systems
Many applications make substantial performance demands upon the computer systems upon which those applications are deployed. In this context, performance is frequently multidimensional, including resource efficiency, power, execution speed (which can be quantified via elapsed run time, data throughput, or latency), and so on. Modern computing platforms exploit parallelism and architectural diversity (e.g., co-processors such as graphics engines and/or reconfigurable logic) to achieve the desired performance goals. This course addresses the practical aspects of achieving high performance on modern computing platforms. This includes questions ranging from how the computing platform is designed to how applications and algorithms expressed to exploit the platform's properties. Particular attention is given to the role of application development tools. Prerequisite: familiarity with software development in Linux preferred, graduate standing or permission of instructor. Credit 3 units. EN: BME T, TU
E81 CSE 567M Computer Systems Analysis
A comprehensive course on performance analysis techniques. The topics include common mistakes, selection of techniques and metrics, summarizing measured data, comparing systems using random data, simple linear regression models, other regression models, experimental designs, 2^“k experimental designs, factorial designs with replication, fractional factorial designs, one factor experiments, two factor full factorial design w/o replications, two factor full factorial designs with replications, general full factorial designs, introduction to queueing theory, analysis of single queues, queueing networks, operational laws, mean-value analysis, time series analysis, heavy tailed distributions, self-similar processes, long-range dependence, random number generation, analysis of simulation results, and art of data presentation. Prerequisites: CSE 260M.
Credit 3 units. EN: BME T, TU

E81 CSE 569S Advanced IoT, Real-Time, and Embedded Systems Security
The aim of this course is to provide students with knowledge and hands-on experience in understanding the security techniques and methods needed for IoT, real-time, and embedded systems. Students complete an independent research project which will involve synthesizing multiple security techniques and applying them to an actual IoT, real-time, or embedded system or device. Credit 3 units. EN: TU

E81 CSE 570S Recent Advances in Networking
This course covers the latest advances in networking. The topics include networking trends, data center network topologies, data center Ethernet, carrier IP, Multiprotocol Label Switching (MPLS), carrier Ethernet, virtual bridging, LAN extension and virtualization using Layer 3 protocols, virtual routing protocols, the "Internet of things," data link layer and management protocols for the internet of things, networking layer protocols for the internet of things, 6LoWPAN, RPL, messaging protocols for the internet of things, MQTT, OpenFlow, software-defined networking (SDN) network function virtualization (NFV), big data, networking issues for big data, network configuration, and data modeling, NETCONF, YIN, YANG, BEEP, and UML.
Prerequisite: CSE 473S or equivalent.
Credit 3 units. EN: BME T, TU

E81 CSE 571S Network Security
This course covers principles and techniques in securing computer networks. Real world examples will be used to illustrate the rationales behind various security designs. There are three main components in the course, preliminary cryptography, network protocol security and network application security. Topics include IPsec, SSL/TLS, HTTPS, network fingerprinting, network malware, anonymous communication, and blockchain. The class project allows students to take a deep dive into the topic of choice in network security. Prerequisite: CSE 473S.
Credit 3 units. EN: BME T, TU

E81 CSE 574S Wireless and Mobile Networking
First course in wireless networking providing a comprehensive treatment of wireless data and telecommunication networks. Topics include recent trends in wireless and mobile networking, wireless coding and modulation, wireless signal propagation, IEEE 802.11a/b/g/n/ac wireless local area networks, 60 GHz millimeter wave gigabit wireless networks, vehicular wireless networks, while spaces, IEEE 802.22 regional area networks, Bluetooth and Bluetooth Smart, wireless personal area networks, wireless protocols for the "internet of things," ZigBee, cellular networks: 1G/2G/3G, LTE, LTE-Advanced, and 5G.
Prerequisites: CSE 473S or permission of the instructor.
Credit 3 units. EN: BME T, TU

E81 CSE 581T Approximation Algorithms
Numerous optimization problems are intractable to solve optimally. The intractability of a problem could come from the problem's computational complexity, for instance the problem is NP-Hard, or other computational barriers. To cope with the inability to find an optimal algorithm, one may desire an algorithm that is guaranteed to return a solution that is comparable to the optimum. Such an algorithm is known as an approximation algorithm. Approximation algorithms are a robust way to cope with intractability, and they are widely used in practice or are used to guide the development of practical heuristics. The area of approximation algorithms has developed a vast theory, revealing the underlying structure of problems as well as their different levels of difficulty. The majority of this course will focus on fundamental results and widely applicable algorithmic and analysis techniques for approximation algorithms. Prerequisite: CSE 347.
Credit 3 units. EN: BME T, TU

E81 CSE 584A Algorithms for Biosequence Comparison
This course surveys algorithms for comparing and organizing discrete sequential data, especially nucleic acid and protein sequences. Emphasis is on tools to support search in massive biosequence databases and to perform fundamental comparison tasks such as DNA short-read alignment. Prerequisites: CSE 347 or instructor's permission. These techniques are also of interest for more general string processing and for building and mining textual databases. Algorithms are presented rigorously, including proofs of correctness and running time where feasible. Topics include classical string matching, suffix array string indices, space-efficient string indices, rapid inexact matching by filtering (including BLAST and related tools), and alignment-free algorithms. Students complete written assignments and implement advanced comparison algorithms to address problems in bioinformatics. This course does not require a biology background. Prerequisites: CSE 347 or instructor's permission. Revised: 2019-02-21
Credit 3 units. EN: BME T, TU

E81 CSE 585T Sparse Modeling for Imaging and Vision
Sparse modeling is at the heart of modern imaging, vision, and machine learning. It is a fascinating new area of research that seeks to develop highly effective data models. The core idea in sparse modeling theory is a novel redundant transform, where the number of transform coefficients is larger compared to the original data dimension. Together with redundancy comes an opportunity of seeking the sparsest possible representation, or the one with the fewest non-zeros. This core idea leads to a series of beautiful theoretical and practical results with many applications such as regression, prediction, restoration, extrapolation, compression, detection, and recognition. In this course, we will explore sparse modeling by covering theoretical as well as algorithmic aspects with applications in computational imaging and computer vision. Prerequisites: ESE 318, Math 233, Math 309, Math 429, or equivalents. Coding with MATLAB or Python.
Credit 3 units. EN: BME T, TU
E81 CSE 586A Analysis of Imaging Data
This course focuses on an in-depth study of advanced topics and interests in image data analysis. Students will learn about hardcore imaging techniques and gain the mathematical fundamentals needed to build their own models for effective problem solving. Topics of deformable image registration, numerical analysis, probabilistic modeling, data dimensionality reduction, and convolutional neural networks for image segmentation will be covered. The main focus might change from semester to semester. Prerequisites: Math 309, ESE 326, and CSE 247. Credit 3 units.

E81 CSE 587A Algorithms for Computational Biology
This course is a survey of algorithms and mathematical methods in biological sequence analysis (with a strong emphasis on probabilistic methods) and systems biology. Sequence analysis topics include introduction to probability, probabilistic inference in missing data problems, hidden Markov models (HMMs), profile HMMs, sequence alignment, and identification of transcription-factor binding sites. Systems biology topics include discovery of gene regulatory networks, quantitative modeling of gene regulatory networks, synthetic biology, and (in some years) quantitative modeling of metabolism. Prerequisites: CSE 131 or CSE 501N. Credit 3 units.

E81 CSE 591 Introduction to Graduate Study in CSE
Introduces students to the different areas of research conducted in the department. Provides an introduction to research skills, including literature review, problem formulation, presentation, and research ethics. Lecture and discussion are supplemented by exercises in the different research areas and in critical reading, idea generation, and proposal writing. Credit 3 units.

E81 CSE 598 Master’s Project
Students electing the project option for their master's degree perform their project work under this course. In order to successfully complete this course, students must defend their project before a three-person committee and present a 2-3 page extended abstract. Prerequisite: permission of adviser and submission of a research proposal form. Credit variable, maximum 6 units.

E81 CSE 599 Master’s Research
Students electing the thesis option for their master's degree perform their thesis research under this course. In order to successfully complete a master's thesis, students must enroll in 6 units of this course typically over the course of two consecutive semesters, produce a written thesis, and defend the thesis before a three-person committee. Prerequisite: permission of adviser and submission of a research proposal form. Credit variable, maximum 6 units.