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 and medicine, music and 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, two undergraduate degrees, combined undergraduate and graduate programs, as well as undergraduate research opportunities. Each academic program can be tailored to a student’s individual needs.

The breadth of Computer Science & 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. While in 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 feed information which 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 in one of these categories: S for software systems, M for machines (hardware), T for theory, and 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 which can be taken as requirements or electives for either of the undergraduate degree programs. We offer a Bachelor of Science in Computer Science (BSCS), a Bachelor of Science in Computer Engineering (BSCoE), or a second major in computer science. In addition to these three 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 design an individualized program, monitor a student’s progress, and consult about curriculum and career options. Additional information can be found on our 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.

Bachelor of Science in Computer Science: 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.

Bachelor of Science in Computer Engineering: 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.

**Second Major in Computer Science:** The second major ([https://cse.wustl.edu/undergraduate/programs/Pages/second-major.aspx](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 second major in computer science programs. Students interested in the pre-medical option should refer to the School of Engineering & Applied Science ([http://bulletin.wustl.edu/undergrad/engineering/#premedicaleducation](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 (admissions@cse.wustl.edu) or call 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](http://bulletin.wustl.edu/undergrad/engineering/requirements)) defined by the School of Engineering & Applied Science (SEAS), 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 requirement. 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 upon a student's cumulative GPA, calculated at the end of the junior year, will be awarded automatically in the student's final year of study. Students are classified as graduate students in their final year of study, and their tuition charges are at the graduate student rate. For information regarding 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](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 at the 500 level during their first four years. Students must 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. Late applications will be accepted after October 1 if submitted prior to the end of the applicant's senior year. Late applicants either have a reduced scholarship scale or, in some cases, may no longer be eligible for scholarship.

**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 the Olin Business School ([http://bulletin.wustl.edu/undergrad/business](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 Computer Science I, CSE 132 Computer Science II, 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 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, BSCS or BScCoE. 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 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 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 (http://careercenter.wustl.edu/tools/careerdevelopment/Pages/engineeringcoop.aspx) that is part of the Career Center in the Danforth University Center, Suite 110.

Research: Participating in an undergraduate research project 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. Visit the CSE website (https://cse.wustl.edu/undergraduate/Pages/undergraduate-research.aspx) for details.

Study Abroad: Students in the 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 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 UK, 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 SEAS 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 Computer Science I. 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 (associatechair@cse.wustl.edu).

Phone: 314-935-6160
Website: https://cse.wustl.edu/undergraduate/programs

Faculty

Chair

Roch Guérin (https://engineering.wustl.edu/Profiles/Pages/Roch-Gu%C3%A9rin.aspx)
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/Profiles/Pages/Sanjoy-Baruah.aspx)
PhD, The 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/Profiles/Pages/Aaron-Bobick.aspx)
James M. McKelvey Professor and Dean
PhD, Massachusetts Institute of Technology
Computer vision, graphics, human-robot collaboration

Michael R. Brent (https://engineering.wustl.edu/Profiles/Pages/Michael-Brent.aspx)
Henry Edwin Sever Professor of Engineering
PhD, Massachusetts Institute of Technology
Systems biology, computational and experimental genomics, mathematical modeling, algorithms for computational biology, bioinformatics

Jeremy Buhler (https://engineering.wustl.edu/Profiles/Pages/Jeremy-Buhler.aspx)
PhD, Washington University
Computational biology, genomics, algorithms for comparing and annotating large biosequences
Roger D. Chamberlain (https://engineering.wustl.edu/Profiles/Pages/Roger-Chamberlain.aspx)
DSc, Washington University
Computer engineering, parallel computation, computer architecture, multiprocessor systems

Yixin Chen (https://engineering.wustl.edu/Profiles/Pages/Yixin-Chen.aspx)
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/Profiles/Pages/Patrick-Crowley.aspx)
PhD, University of Washington
Computer and network systems, network security

Ron K. Cytron (https://engineering.wustl.edu/Profiles/Pages/Ron-Cytron.aspx)
PhD, University of Illinois at Urbana-Champaign
Programming languages, middleware, real-time systems

Christopher D. Gill (https://engineering.wustl.edu/Profiles/Pages/Christopher-Gill.aspx)
DSc, Washington University
Distributed real-time embedded systems, middleware, formal models and analysis of concurrency and timing

Raj Jain (https://engineering.wustl.edu/Profiles/Pages/Raj-Jain.aspx)
PhD, Harvard University
Wireless networks, network security, next generation internet, sensor networks, telecommunications networks, performance analysis, traffic management, quality of service

Tao Ju (https://engineering.wustl.edu/Profiles/Pages/Tao-Ju.aspx)
PhD, Rice University
Computer graphics, visualization, mesh processing, medical imaging and modeling

Chenyang Lu (https://engineering.wustl.edu/Profiles/Pages/Chenyang-Lu.aspx)
Fullgraf Professor in the Department of Computer Science & Engineering
PhD, University of Virginia
Real-time and embedded systems, wireless sensor networks, mobile computing

Weixiong Zhang (https://engineering.wustl.edu/Profiles/Pages/Weixiong-Zhang.aspx)
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/Profiles/Pages/Kunal-Agrawal.aspx)
PhD, Massachusetts Institute of Technology
Parallel computing, cyber-physical systems & sensing, theoretical computer science

Sanmay Das (https://engineering.wustl.edu/Profiles/Pages/Sanmay-Das.aspx)
PhD, Massachusetts Institute of Technology
Design of algorithms for complex environments, computational social science, machine learning

Caitlin Kelleher (https://engineering.wustl.edu/Profiles/Pages/Caitlin-Kelleher.aspx)
Hugo F. & Ina Champ Urbauer Career Development Associate Professor
PhD, Carnegie Mellon University
Human-computer interaction, programming environments, and learning environments

William D. Richard (https://engineering.wustl.edu/Profiles/Pages/William-Richard.aspx)
PhD, University of Missouri-Rolla
Ultrasonic imaging, medical instrumentation, computer engineering

Assistant Professors

Ayan Chakrabarti (https://engineering.wustl.edu/Profiles/Pages/Ayan-Chakrabarti.aspx)
PhD, Harvard University
Computer vision computational photography, machine learning

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

Chien-Ju Ho (https://engineering.wustl.edu/Profiles/Pages/Chien-Ju-Ho.aspx)
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/Profiles/Pages/Brendan-Juba.aspx)
PhD, Massachusetts Institute of Technology
Theoretical approaches to artificial intelligence founded on computational complexity theory and theoretical computer science more broadly construed
**Ulugbek Kamilov**
PhD, École Polytechnique Fédérale de Lausanne, Switzerland
Computational imaging, image and signal processing, machine learning, and optimization

**Brian Kocoloski**
PhD, University of Pittsburgh
Scalable parallel computing, cloud computing, operating systems, virtualization

**Angelina Lee**
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

**Benjamin Moseley**
PhD, University of Illinois at Urbana-Champaign
Design and analysis of algorithms, online and approximation algorithms, parallel computing, large data analysis, green computing and algorithmic applications

**Alvitta Ottley**
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

**Douglas Shook**
MS, Washington University
Imaging sensor design, compiler design and optimization

**William Siever**
PhD, Missouri University of Science and Technology
Principal Lecturer

**William Yeoh**
PhD, University of Southern California
Artificial intelligence, multi-agent systems, distributed constraint optimization, planning and scheduling

**Senior Professors**

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

**Mark A. Franklin**
Hugo F. and Ina Champ Urbauer Professor of Engineering
PhD, Carnegie Mellon University
Computer architecture, systems analysis and parallel processing, storage systems design

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

**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 refer to the following sections for information about the Bachelor of Science in Computer Science (p. 5), the Bachelor of Science in Computer Engineering (p. 6), and the second major in computer science (p. 7).

**Bachelor of Science in Computer Science**

The Bachelor of Science in Computer Science (BSCS) is designed for students planning a career in computing. Students working toward a BSCS degree must meet all requirements for an applied science degree (http://bulletin.wustl.edu/undergrad/engineering/requirements) from the School of Engineering & Applied Science. In addition, there are the following departmental course requirements:
Computer Science Core Requirements

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSE 131</td>
<td>Computer Science I</td>
<td>3</td>
</tr>
<tr>
<td>CSE 132</td>
<td>Computer Science II</td>
<td>3</td>
</tr>
<tr>
<td>CSE 240</td>
<td>Logic and Discrete Mathematics</td>
<td>3</td>
</tr>
<tr>
<td>or Math 310</td>
<td>Foundations for Higher Mathematics</td>
<td></td>
</tr>
<tr>
<td>CSE 247</td>
<td>Data Structures and Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>CSE 332S</td>
<td>Object-Oriented Software Development Laboratory</td>
<td></td>
</tr>
<tr>
<td>CSE 347</td>
<td>Analysis of Algorithms</td>
<td>3</td>
</tr>
</tbody>
</table>

Total Units: 18

Each of these core courses must be passed with a grade of C- or better.

Computer Science Technical Elective Requirements:

At least 24 additional units in computer science or computer science-related courses with an S, M, T or A suffix of which at least one must be a systems (S) course and at least one must be a machine (M) or application (A) course. Students may use up to 6 units of approved independent work (CSE 400E, CSE 497-CSE 499) as part of their computer science electives. Such independent work can be classified as S, M or A with approval.

There are several technical elective course sequences (https://cse.wustl.edu/undergraduate/programs/Pages/bs-in-computer-science.aspx) described on the department webpage that are recommended for students whose interests are in particular areas of computing.

Math Requirements:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math 131</td>
<td>Calculus I</td>
<td>3</td>
</tr>
<tr>
<td>Math 132</td>
<td>Calculus II</td>
<td>3</td>
</tr>
<tr>
<td>Math 233</td>
<td>Calculus III</td>
<td>3</td>
</tr>
<tr>
<td>Math 309</td>
<td>Matrix Algebra</td>
<td>3</td>
</tr>
<tr>
<td>or ESE 318</td>
<td>Engineering Mathematics A</td>
<td></td>
</tr>
<tr>
<td>ESE 326</td>
<td>Probability and Statistics for Engineering</td>
<td>3</td>
</tr>
<tr>
<td>or Math 3200</td>
<td>Elementary to Intermediate Statistics and Data Analysis</td>
<td></td>
</tr>
<tr>
<td>or QBA 120 &amp; QBA 121</td>
<td>Managerial Statistics I and Managerial Statistics II</td>
<td></td>
</tr>
<tr>
<td>or Psych 300</td>
<td>Introduction to Psychological Statistics</td>
<td></td>
</tr>
</tbody>
</table>

Total Units: 15

Bachelor of Science in Computer Engineering

Computer Engineering encompasses studies of hardware, software, and systems issues that arise in the design, development, and application of computer systems. Computer engineers are particularly well suited to address the particular challenges that exist as computing systems interact with the real, physical world. This includes sensing, actuation, timing, security, and computing systems with widely varying form factors (ranging from servers to mobile devices to the Internet of Things (IoT)).

Students working toward a BScCoE degree must meet all requirements for an engineering degree (http://bulletin.wustl.edu/undergrad/engineering/requirements) from the School of Engineering & Applied Science. Required courses and technical electives cannot be taken on a pass/fail basis:

Common Studies Program Requirements:

Computer engineering students must meet the common studies program requirements, which are required of all students pursuing an engineering degree. These requirements, as they pertain to BScCoE students specifically, are described below:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Math 131</td>
<td>Calculus I</td>
<td>3</td>
</tr>
<tr>
<td>Math 132</td>
<td>Calculus II</td>
<td>3</td>
</tr>
<tr>
<td>Math 217</td>
<td>Differential Equations</td>
<td>3</td>
</tr>
<tr>
<td>Math 233</td>
<td>Calculus III</td>
<td>3</td>
</tr>
<tr>
<td>Physics 117A</td>
<td>General Physics I</td>
<td>4</td>
</tr>
<tr>
<td>or Physics 197</td>
<td>Physics I</td>
<td></td>
</tr>
</tbody>
</table>
Upon completing a course in the calculus sequence (Math 131-Math 132-Math 233) with a grade of C+ or better, the student may apply to receive credit for the preceding courses in the calculus sequence by following the math department’s back credit policy (http://college.artsci.wustl.edu/policies/placement_credit/backcredit).

The Natural Sciences requirement is for 3 units designated NSM (Natural Sciences and Mathematics) from any of the following departments: Anthropology, Biology, Chemistry, Earth and Planetary Sciences, Environmental Studies, or Physics. These courses must be completed with a grade of C- or better.

The College Writing and Humanities and Social Sciences requirements are those required of all students in the School of Engineering & Applied Science.

**Computer Engineering Core Requirements:**

The following courses are required of all computer engineering students:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSE 131</td>
<td>Computer Science I</td>
<td>3</td>
</tr>
<tr>
<td>CSE 132</td>
<td>Computer Science II</td>
<td>3</td>
</tr>
<tr>
<td>ESE 230</td>
<td>Introduction to Electrical and Electronic Circuits</td>
<td>4</td>
</tr>
<tr>
<td>ESE 232</td>
<td>Introduction to Electronic Circuits</td>
<td>3</td>
</tr>
<tr>
<td>CSE 240</td>
<td>Logic and Discrete Mathematics</td>
<td>3</td>
</tr>
<tr>
<td>or Math 310</td>
<td>Foundations for Higher Mathematics</td>
<td></td>
</tr>
<tr>
<td>CSE 247</td>
<td>Data Structures and Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>CSE 260M</td>
<td>Introduction to Digital Logic and Computer Design</td>
<td>3</td>
</tr>
<tr>
<td>ESE 326</td>
<td>Probability and Statistics for Engineering</td>
<td>3-6</td>
</tr>
<tr>
<td>or Math 3200</td>
<td>Elementary to Intermediate Statistics and Data Analysis</td>
<td></td>
</tr>
<tr>
<td>or QBA 120 &amp; QBA 121</td>
<td>Managerial Statistics I and Managerial Statistics II</td>
<td></td>
</tr>
<tr>
<td>or Psych 300</td>
<td>Introduction to Psychological Statistics</td>
<td></td>
</tr>
<tr>
<td>CSE 361S</td>
<td>Introduction to Systems Software</td>
<td>3</td>
</tr>
<tr>
<td>CSE 362M</td>
<td>Computer Architecture</td>
<td>3</td>
</tr>
<tr>
<td>CSE 422S</td>
<td>Operating Systems Organization</td>
<td>3</td>
</tr>
<tr>
<td>CSE 462M</td>
<td>Computer Systems Design</td>
<td>3</td>
</tr>
</tbody>
</table>

or CSE 465M Digital Systems Laboratory

Total Units 37-40

Each of these core courses must be passed with a grade of C- or better.

**Computer Engineering Focus Area:**

At least 18 units of technical electives, drawn from either of the following:

- CSE courses with suffix S, M, T or A,
- ESE courses at the 300-level or higher.

The above can include courses at the graduate level; however, they must still meet one of the two criteria above. Up to 6 units of Independent Study (CSE 400E, CSE 497-CSE 499, ESE 400, ESE 497) can count toward technical electives. There is no limit as to how many independent study courses can count toward the general 120 units.

**The Second Major in Computer Science**

The second major 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 also is well suited for students planning careers in medicine, law, business, architecture and fine arts. The requirements are as follows. There are no additional distribution or unit requirements for the second major.

**Computer Science Core Requirements:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSE 131</td>
<td>Computer Science I</td>
<td>3</td>
</tr>
<tr>
<td>CSE 132</td>
<td>Computer Science II</td>
<td>3</td>
</tr>
<tr>
<td>CSE 240</td>
<td>Logic and Discrete Mathematics</td>
<td>3</td>
</tr>
<tr>
<td>or Math 310</td>
<td>Foundations for Higher Mathematics</td>
<td></td>
</tr>
<tr>
<td>CSE 247</td>
<td>Data Structures and Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>CSE 332S</td>
<td>Object-Oriented Software</td>
<td>3</td>
</tr>
<tr>
<td>or Math 3200</td>
<td>Elementary to Intermediate Statistics and Data Analysis</td>
<td></td>
</tr>
<tr>
<td>or QBA 120 &amp; QBA 121</td>
<td>Managerial Statistics I and Managerial Statistics II</td>
<td></td>
</tr>
<tr>
<td>or Psych 300</td>
<td>Introduction to Psychological Statistics</td>
<td></td>
</tr>
<tr>
<td>CSE 347</td>
<td>Analysis of Algorithms</td>
<td>3</td>
</tr>
</tbody>
</table>

Total Units 18

Each of these core courses must be passed with a grade of C- or better.

**Computer Science Electives:**

At least 24 units of computer science electives, selected from any CSE courses with an S, M, T or A suffix.

**Math Requirement:**
Calculus (Math 131) and Probability (ESE 326 or Math 3200, or the sequence QBA 120-QBA 121).

Minors

Please refer to the following sections for information about the minor in computer science (p. 8) and the minor in bioinformatics (p. 8).

The Minor in Computer Science

If a student's goal is a basic foundation in computer science for application to another field, but they are not planning a career as a practicing computer scientist, the minor in computer science is a good choice. The minor consists of five CSE courses, including two core courses and three electives. The core courses provide an introduction to computer science concepts and problem-solving techniques. The electives offer flexibility to integrate computer science studies with a major area. Students select the courses that are most important to them, whether their interests are in fine arts, linguistics, psychology, philosophy, law, business, medicine, the natural sciences or anything else.

Units required: 15

Required courses:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSE 131</td>
<td>Computer Science I</td>
<td>3</td>
</tr>
<tr>
<td>CSE 247</td>
<td>Data Structures and Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>Total Units</td>
<td></td>
<td>6</td>
</tr>
</tbody>
</table>

Elective courses:

Any three additional CSE courses selected from CSE 132 Computer Science II, CSE 240 Logic and Discrete Mathematics, CSE 347 Analysis of Algorithms, or any CSE course with a T, S, M or A suffix.

Additional Information

All courses used for the computer science minor must be taken for a grade, and the student must earn a C- or better. Should the student decide to go further in the field, all courses in the computer science minor can be used toward a second major in computer science or a degree in computer science or computer engineering.

The Minor in Bioinformatics

Mindful of the emerging opportunities at the interface of biology and computer science, the Department of Biology and Department of Computer Science & Engineering have fashioned the minor in bioinformatics that serves students from both departments as well as other students from the natural sciences and engineering with an interest in this field.

Units required: 23-24 units as described below

Core courses:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biol 2960</td>
<td>Principles of Biology I</td>
<td>4</td>
</tr>
<tr>
<td>Biol 2970</td>
<td>Principles of Biology II</td>
<td>4</td>
</tr>
<tr>
<td>CSE 131</td>
<td>Computer Science I</td>
<td>3</td>
</tr>
<tr>
<td>CSE 247</td>
<td>Data Structures and Algorithms</td>
<td>3</td>
</tr>
<tr>
<td>Math 2200</td>
<td>Elementary Probability and Statistics</td>
<td>3</td>
</tr>
<tr>
<td>or Math 3200</td>
<td>Elementary to Intermediate Statistics and Data Analysis</td>
<td>3</td>
</tr>
<tr>
<td>or ESE 326</td>
<td>Probability and Statistics for Engineering</td>
<td>3</td>
</tr>
<tr>
<td>Total Units</td>
<td></td>
<td>17</td>
</tr>
</tbody>
</table>

Advanced biology elective: Choose one of:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Biol 3492</td>
<td>Laboratory Experiments with Eukaryotic Microbes</td>
<td>3</td>
</tr>
<tr>
<td>Biol 4181</td>
<td>Population Genetics</td>
<td>3</td>
</tr>
<tr>
<td>Biol 4342</td>
<td>Research Explorations in Genomics</td>
<td>4</td>
</tr>
<tr>
<td>Biol 437</td>
<td>Laboratory on DNA Manipulation</td>
<td>4</td>
</tr>
</tbody>
</table>

CSE elective: Choose one of:

<table>
<thead>
<tr>
<th>Code</th>
<th>Title</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSE 514A</td>
<td>Data Mining</td>
<td>3</td>
</tr>
<tr>
<td>CSE 584A</td>
<td>Algorithms for Biosequence Comparison</td>
<td>3</td>
</tr>
<tr>
<td>or Biol 5504</td>
<td>Algorithms for Biosequence Comparison</td>
<td>3</td>
</tr>
<tr>
<td>CSE 587A</td>
<td>Algorithms for Computational Biology</td>
<td>3</td>
</tr>
</tbody>
</table>

Additional Information

It is anticipated that for those students majoring in biology or CSE, some portion of the introductory sequence will overlap with courses required for the major, and these courses will be applicable to both the major and the minor. Upper-level courses in Biology and CSE used to fulfill the minor may not be used to fulfill another major or minor in Arts & Sciences. A minimum grade of C- is required for all courses to count toward the minor.

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 104 Web Development
This comprehensive course does not assume prior programming background or web design experience. Explores elementary principles that go into designing, creating, and publishing an effective website. Topics include the production process, design metaphors, interface/information design, page layout concepts, graphics preparation, color theory, development tools, HTML, style sheets, basic scripting techniques, search engine optimization and site maintenance/marketing strategies. Credit 3 units. EN: TU

E81 CSE 131 Computer Science I
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 Computer Science II
CSE 132 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. EN: TU

E81 CSE 141 The Digital Society
Our modern digital society is both exciting and challenging. As the effectiveness of computing advances, and digital technologies like the internet and algorithmic decision-making affect and penetrate more and more aspects of our lives, we face extraordinary opportunities and equally extraordinary challenges. Computer driven automation increases our quality of life but eradicates our jobs! The internet and smartphones keep us connected but subject us to growing corporate, government and criminal surveillance! In this course, co-taught by professors from the Schools of Engineering and Law, we will examine the fundamental technical underpinnings of digital society and its consequences. We will discuss "Welcome to the Future," "The Future of Jobs," and "The Future of Humans." All aspects of life are evolving rapidly in our digital society, and we will draw on expert and engaging guest speakers from all seven schools of Washington University and intellectual leaders from beyond our campus to share their perspectives and insights. This course will help students to perceive the modern world in new ways in order to better understand how technological shifts are changing and challenging notions of individual and collective prosperity. Our goal is to give students both the technical understanding of how our new technologies work and the critical skills to evaluate them for themselves as citizens and leaders of our new digital society. Same as I50 InterD 141 Credit 3 units.

E81 CSE 222S Internet of Things
For a very long time, the Things in our world have lived relatively lonely and single-purposed lives. With the advent of the Internet of Things, we can address, control, and interconnect these formerly isolated devices to create new and interesting applications. In this course we study Bluetooth Low Energy, one of the fundamental networking technologies behind Internet-of-Things devices, and Appcessories, which include smart watches, health monitors, toys and appliances. In addition to learning about the Bluetooth Low Energy protocol and network stack, students gain hands-on experience developing multi-platform solutions that control and communicate with Things using an accompanying app on a mobile device. Students apply their knowledge and skill to develop a project of their choosing using topics from the course. Prerequisite: CSE 132. 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. Collaboration skills are developed through work performed in small teams. Prerequisite: CSE 131 or equivalent. Credit 3 units. EN: TU

E81 CSE 232S 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. Prerequisites: CSE 132. 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
After a formal presentation of a topic, students develop a related interest. The course implements an interactive studio format: able to synthesize the course material into a project of their own.

E81 CSE 247 Data Structures and Algorithms
Study of fundamental algorithms, data structures, and their effective use in a variety of applications. Emphasizes 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 240.
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 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 247 and ESE 326.
Credit 3 units. EN: 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 the AngularJS web framework, 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 a formal presentation of a topic, students develop a related project under the supervision of the instructor. Prerequisite: CSE 131.
Credit 3 units. EN: 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 247.
Credit 3 units. EN: 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 247.
Credit 3 units. EN: 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. Prerequisite: CSE 247.
Credit 3 units. EN: TU

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 enable 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 132.
Credit 3 units. EN: 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: 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
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: TU
is provided at the end of the course for students to work on a project of their own interest. Prerequisites: 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 of games through analysis of current games. Students have the opportunity to explore additional topics including graphics, artificial intelligence, networking, physics, and user interface design through their game project. Prerequisite: CSE 3xxS or 4xxS. 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-conversion, 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 3325 and Math 309. 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 the use of FPGA design methodologies. Students use a commercial CAE/CAD system for VHDL-based design and simulation while designing a selected computation system. Prerequisites: CSE 361S and CSE 362M. 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: CSE 260M and CSE 361S. Credit 3 units. EN: TU

E81 CSE 467S Embedded Computing Systems
Introduces the issues, challenges and methods for designing embedded computing systems — systems designed to serve a particular application, which incorporate the use of digital processing devices. Examples of embedded systems include PDAs, cellular phones, appliances, game consoles, automobiles and iPod. 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 perform a course project on a real wireless sensor network testbed. Prerequisite: CSE 361S. 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. Prerequisites: CSE 132 and 247. Credit 3 units. EN: TU

E81 CSE 483M Introduction to Electronic Instrumentation Design
This course covers the principles underlying the design of electronic instrumentation systems. As a case study we will design and analyze a complete instrumentation amplifier integrated circuit and discuss topics related to signal conditioning, amplification, filtering, biasing, data conversion, interface design, mixed-signal layout and mixed-signal simulation. Students are expected to work with state-of-the-art circuit simulation and layout tools for their homework and project assignments. Prerequisites: CSE 260 and ESE 232. Credit 3 units. EN: 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 exploration and extension of theoretical structures, or may pivot around the design/development of solutions for particular applications drawn from areas throughout the University and/or 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 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 exploration and extension of theoretical structures, or may pivot around the design/development of solutions for particular applications drawn from areas throughout the University and/or 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 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
Credit variable, maximum 3 units.

E81 CSE 501N Programming Concepts and Practice
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 Fundamentals of Computer Science
Study of fundamental algorithms, data structures, and their effective use in a variety of applications. Emphasizes 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 240.
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 the AngularJS web framework, 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 a formal presentation of a topic, students develop a related project under the supervision of the instructor. Prerequisite: CSE 131.
Same as E81 CSE 330S
Credit 3 units. EN: TU

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 247.
Same as E81 CSE 332S
Credit 3 units. EN: 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: TU

E81 CSE 506M Principle and Methods of Micro- and Nanofabrication
A hands-on introduction to the fundamentals of micro- and nanofabrication processes with emphasis on cleanroom practices. The physical principles of oxidation, optical lithography, thin film deposition, etching and metrology methods will be discussed, demonstrated and practiced. Students will be trained in cleanroom concepts and safety protocols. Sequential microfabrication processes involved in the manufacture of microelectronic and photonic devices will be shown. Training in imaging and characterization of micro- and nanostructures will be provided. Prerequisites: graduate or senior standing or permission of the instructor.
Same as E37 MEMS 5611
Credit 3 units. EN: TU

E81 CSE 507A Technology Entrepreneurship
This is a course for students who plan to be, or 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-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 include: business ethics, opportunity assessment, team formation, financing, intellectual property and university technology transfer. The course features significant participant and guest instruction from experienced practitioners. Prerequisites: none.
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 347, ESE 326, Math 233.
Credit 3 units.

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: 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 cameras 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 learning. Prerequisites: CSE 247 and ESE 326 (or Math 320) or their equivalent, or permission of the instructor.
Credit 3 units. EN: 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: TU

E81 CSE 516A 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, or permission of instructor. Some prior exposure to artificial intelligence, machine learning, game theory, and microeconomics may be helpful, but is not required.
Credit 3 units. EN: 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, Gaussian processes), 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 247, CSE 417T, ESE 326, Math 233 and Math 309. The instructor will hold a take-home placement exam (on basic mathematical knowledge) for all students currently enrolled and on the waitlist. The exam will be due on the first day of class. Only students who pass the placement exam will be enrolled in the course.
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, 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 517A or 511A or 7100.
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 422S.
Credit 3 units. EN: TU
E81 CSE 521S Wireless Sensor Networks
Dense collections of smart sensors networked to form self-configuring pervasive computing systems provide a basis for 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 studies a large number of research papers that deal with various aspects of wireless sensor networks. Students perform a project on a real wireless sensor network composed of tiny devices each consisting of sensors, a radio transceiver and a microcontroller. Prerequisite: CSE 422S. Credit 3 units. EN: TU

E81 CSE 522S Advanced Operating Systems
This course explores core OS abstractions, mechanisms and policies and how they impact support for general purpose, embedded and real-time operating environments. How to evaluate, modify and optimize the use of kernel-level resources is covered hands-on and in detail, including CPU and I/O scheduling, memory management, and interprocess communication. Prerequisite: CSE 422S. Credit 3 units. EN: 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: 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 531S Theory of Compiling and Language Translation
Algorithms and intermediate representations for automatic program analysis are examined, with an emphasis on practical methods and efficient engineering of program optimization and transformations. The course includes a thorough treatment of monotone data flow frameworks: a mathematical model in which most optimization problems can be specified and solved. The course primarily covers optimizations that are applicable to any target architecture; however, optimizations specific to parallel, distributed and storage-hierarchical systems also are discussed. Prerequisite: CSE 431S or CSE 425S. Credit 3 units. EN: TU

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 C++11 concurrency and synchronization features and software architecture patterns. Prerequisites: CSE 332S/CSE 504N or graduate standing and strong familiarity with C++; and CSE 422S or CSE 522S. Credit 3 units. EN: TU

E81 CSE 536S Distributed System Design: Models and Languages
Modern computing environments are highly distributed. This has been the result of major advances in networking technology and their rapid assimilation by a society that functions in a highly distributed and decentralized manner. The goal of this course is to familiarize students with basic concepts, models and languages that shaped recent developments in distributed computing. The focus is on exploring new ways of thinking about computing and communication that made the development of distributed software systems possible. Competing concepts and design strategies will be examined both from a theoretical and a practical perspective. Prerequisites: CSE 240 and CSE 247. Credit 3 units. EN: TU

E81 CSE 538T Modeling and Performance Evaluation of Interconnected Computer Systems
Modern computing systems consist of multiple interconnected components, which 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. The course covers Markov chains and their applications to simple queues, and proceeds to explore more complex systems including server farms and how to optimize their performance through scheduling and task assignment policies. The course includes a brief review of the necessary probability and mathematical concepts. Prerequisite: ESE 326. 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 scheduling, high-level parallel language models, concurrent programming (synchronization and concurrent data structures), algorithms for debugging parallel software, and performance analysis. Prerequisites: CSE 332S and CSE 361S. Credit 3 units. EN: TU
E81 CSE 541T Advanced Algorithms
Provides a broad coverage of fundamental algorithm design
techniques with the focus on developing efficient algorithms for
solving combinatorial and optimization problems. The topics
covered include: greedy algorithms, dynamic programming,
linear programming, NP-completeness, approximation
algorithms, lower bound techniques, and on-line algorithms.
Throughout this 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 247 and 347.
Credit 3 units. EN: TU

E81 CSE 542T Advanced Data Structures and Algorithms
This course is concerned with the design and analysis of efficient
algorithms, focusing principally on algorithms for combinatorial
optimization problems. A key element in the course is the role
of data structures in algorithm design and the use of amortized
complexity analysis to determine how data structures affect
performance. The course is organized around a set of core
problems and algorithms, including the classical network
optimization algorithms, as well as newer and more efficient
algorithms. This core is supplemented by algorithms selected from
the recent technical literature. Prerequisite: CSE 247.
Credit 3 units.

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
quadric-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. Prerequisite: Calculus I and Math 309.
Credit 3 units.

E81 CSE 544T Special Topics in Computer Science Theory
Cake-cutting algorithms consider the division of resources
among a set of participants such that the recipients believe they
have been treated fairly. In some cases, a given resource can
be divided without loss of value, while in other cases, dividing a
resource may lessen its value, perhaps significantly. Notions of
fairness include proportionality, envy-freeness and equitability.
This course is organized around a rich set of fair-division
problems, studying the correctness, complexity, and applicability
of algorithms for solving such problems. The problems and
algorithms studied span millennia and include recent technical
literature.
Credit 3 units.

E81 CSE 546T Computational Geometry
Computational geometry is the algorithmic study of problems that
involve geometric shapes such as points, lines and polygons.
Such problems appear in computer graphics, vision, robotics,
animation, visualization, molecular biology, and geographic
information systems. This course covers data structures that are
unique to geometric computing, such as convex hull, Voronoi
diagram, Delaunay triangulation, arrangement, range searching,
KD-trees, and segment trees. Also covered are algorithms for
polygon triangulation, shortest paths, the post office problem,
and the art gallery problem. Prerequisite: CSE 247.
Credit 3 units.

E81 CSE 547T Introduction to Formal Languages and
Automata
An introduction to the theory of computation, with emphasis on
the relationship between formal models of computation and the
computational problems solvable by those models. Specifically,
this course covers finite automata and regular languages;
Turing machines and computability; and basic measures of
computational complexity and the corresponding complexity
classes. Prerequisite: CSE 247.
Credit 3 units.

E81 CSE 548T Concurrent Systems: Design and Verification
Formerly CSE 563T. Concurrency presents programmers
with unprecedented complexity further exacerbated by our
limited ability to reason about concurrent computations. Yet,
concurrent algorithms are central to the development of software
executing on modern multiprocessors or across computer
networks. This course reviews several important classes
of concurrent algorithms and presents a formal method for
specifying, reasoning about, verifying, and deriving concurrent
algorithms. The selected algorithms are judged to have made
significant contributions to our understanding of concurrency.
Rigorous treatment of the design and programming process is
emphasized. Students entering this course must be familiar with
parallel algorithms and their analysis, cache-coherence
analysis in the work/span model, scheduling algorithms, external
parallel computing. Topics include parallel algorithms and
concurrent algorithms are central to the development of software
networks. This course reviews several important classes
of concurrent programs and to apply systematically and
correctly their formal skills to larger problems. Prerequisite: CSE
247.
Credit 3 units.

E81 CSE 549T Theory of Parallel Systems
The course covers various aspects of parallel programming
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.

E81 CSE 552A Advanced Computer Graphics
This course covers advanced topics in graphics in the areas of
modeling, rendering, volume rendering, image-based rendering
and image processing. Topics include, but are not limited to,
subdivision surfaces, splines, mesh simplification, implicit
or blobby modeling, radiosity, procedural textures, filtering,
BRDFs and procedural modeling. The class has several
structured programming assignments and an optional final group
project. Students are exposed to the wide variety of techniques
available in graphics and also pick one area to study in depth.
Prerequisite: CSE 332S and CSE 452A.
Credit 3 units. EN: TU
E81 CSE 553S Advanced Mobile Robotics
This course covers advanced topics from the theory and practice of mobile robotics. Students read, present and discuss papers from the current research literature. There is a substantial programming project, in which students implement and test ideas from the current research literature on one of the department's research robot platforms. Prerequisites: CSE 550A and strong programming skills (preferably in C++). Credit 3 units. EN: 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.). Prerequisites: CSE 247 and CSE 332 or approval by instructor. Credit 3 units. EN: 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 camera 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. Permission of instructor required to enroll. Credit 3 units. EN: 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 be 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: TU

E81 CSE 557A Information 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 and/or complex data visually to aid comprehension and cognition. In this course, we study the principles for transforming abstract data into effective information visualizations. We also learn how to critique existing visualizations and how to evaluate the systems we build. Readings will include current research papers from the Information Visualization community. Prerequisite: permission of professor (this may change in subsequent semesters). Credit 3 units. EN: TU

E81 CSE 558A Motion Planning
This course studies the general motion planning problem: computing a sequence of motions that transforms a given (initial) arrangement of physical objects to another (goal) arrangement of those objects. Many motion planning methods were developed in the realm of robotics research. For example, a typical problem might be to find a sequence of motions (called a path) to move a robot from one position to another without colliding with any objects in its workspace. However, the general motion planning problem that will be studied arises in many other application domains as well. For example, assembly planning (e.g., finding a valid order for adding the parts when building an engine), mechanical CAD studies (e.g., can you remove a certain part from an engine without taking the engine apart), artificial life simulations (e.g., moving a herd of animals from one location to another), and medicine (e.g., can a drug molecule reach a protein molecule). Prerequisite: CSE 247. Credit 3 units. EN: TU

E81 CSE 559A Computer Vision
Computer vision is the process of automatically extracting information from images and video. This course covers imaging geometry (camera calibration, stereo, and panoramic image stitching), and algorithms for video surveillance (motion detection and tracking), segmentation and object recognition. Final projects for the course will explore challenges in analysis of real-world data. Students with non-standard backgrounds (such as video art, or the use of imaging in physics and biology) are encouraged to contact the instructor. Prerequisites: CSE 247 and linear algebra. Credit 3 units. EN: 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: TU
E81 CSE 561M Computer Systems Architecture II
Advanced techniques in computer system design. Selected topics from: processor and system-on-chip design (multicore organization, system-level integration), run-time systems, memory systems (topics in locality and special-purpose memories), I/O subsystems and devices, systems security, and power considerations. Prerequisite: CSE 560M or permission of instructor.
Credit 3 units. EN: TU

E81 CSE 564M Advanced Digital Systems Engineering
This course focuses on advance sensor design. The class covers various basic analog and digital building blocks that are common in most sensor integrated circuits. The class extensively uses state-of-the-art CAD program Cadence to simulate and analyze various circuit blocks. The first half of the course focuses on analyzing various operational amplifiers, analog filters, analog memory and analog to digital converters. The second half of the course focuses on understanding the basic building blocks of imaging sensors. The class has a final project consisting of designing a smart sensor using Cadence tools. Prerequisites: ESE 232 and CSE 362M.
Credit 3 units. EN: TU

E81 CSE 565M Acceleration of Algorithms in Reconfigurable Logic
Reconfigurable logic, in the form of Field-Programmable Gate Arrays (FPGAs), enables the deployment of custom hardware for individual applications. To exploit this capability, the application developer is required to specify the design at the register-transfer level. This course explores techniques for designing algorithms that are amenable to hardware acceleration as well as provides experience in actual implementation. Example applications are drawn from a variety of fields, such as networking, computational biology, etc. Prerequisites: basic digital logic (CSE 280M) and some experience with a hardware description language (e.g., VHDL or Verilog).
Credit 3 units. EN: TU

E81 CSE 566M 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), etc. 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 are 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: 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 131 and CSE 260M.
Credit 3 units. EN: TU

E81 CSE 568M Imaging Sensors
This course will cover topics on digital imaging sensors including basic operations of silicon photodetectors; CCD and CMOS passive and active sensor operation; temporal and spatial noise in CMOS sensors; spatial resolution and MTF; SNR and dynamic range; high dynamic range architectures and application specific imaging sensors such as polarization imaging and fluorescent imaging sensors. Prerequisites: CSE 260M and ESE 232.
Credit 3 units.

E81 CSE 569M Parallel Architectures and Algorithms
A number of contemporary parallel computer architectures are reviewed and compared. The problems of process synchronization and load balancing in parallel systems are studied. Several selected applications problems are investigated and parallel algorithms for their solution are considered. Selected parallel algorithms are implemented in both a shared memory and distributed memory parallel programming environment. Prerequisites: graduate standing and knowledge of the C programming language.
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, Multi-Protocol Label Switching (MPLS), Carrier Ethernet, Virtual Bridging, LAN Extension and Virtualization using Layer 3 Protocols, Virtual Routing Protocols, Internet of Things (IoT), Datalink Layer and Management Protocols for IoT, Networking Layer Protocols for IoT, 6LoWPAN, RPL, Messaging Protocols for IoT, 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: TU

E81 CSE 571S Network Security
Credit 3 units. EN: TU
E81 CSE 573S Protocols for Computer Networks
An introduction to the design, performance analysis and implementation of existing and emerging computer network protocols. Protocols include multiple access protocols (e.g., CSMA/CD, token ring), internet, working with the internet protocol (IP), transport protocols (e.g., UDP, TCP), high-speed bulk transfer protocols, and routing protocols (e.g., BGP, OSPF). General topics include error control, flow control, packet switching, mechanisms for reliable, ordered and bounded-time packet delivery, host-network interfacing and protocol implementation models. Substantial programming exercises supplement lecture topics. Prerequisite: CSE 473S or permission of the instructor.
Credit 3 units. EN: 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, white spaces, IEEE 802.22 regional area networks, Bluetooth and Bluetooth Smart, wireless personal area networks, wireless protocols for 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: TU

E81 CSE 577M Design and Analysis of Switching Systems
Switching is a core technology in a wide variety of communication networks, including the internet, circuit-switched telephone networks and optical fiber transmission networks. The last decade has been a time of rapid development for switching technology in the internet. Backbone routers with 10 Gb/s links and aggregate capacities of hundreds of gigabits per second are becoming common, and advances in technology are now making multi-terabit routers practical. This course is concerned with the design of practical switching systems and evaluation of their performance and complexity. Prerequisites: CSE 247, 473S and ESE 326.
Credit 3 units. EN: 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: TU

E81 CSE 582T Complexity Theory
An introduction to the quantitative theory of computation with limited resources. The course examines the relative power of limited amounts of basic computational resources, such as time, memory, circuit size and random bits, as well as parallel, nondeterministic, alternating and interactive machine models. Models that capture special kinds of computational problems, such as counting problems or approximate solutions, will also be introduced and related to the standard models. This examination will emphasize surprising relationships between seemingly disparate resources and kinds of computational problems. The course will also discuss some meta-theory, illuminating the weaknesses of standard mathematical techniques of the field against its notorious open conjectures. Prerequisites: CSE 347 and mathematical maturity. CSE 547T recommended.
Credit 3 units. EN: TU

E81 CSE 583A Topics in Computational Molecular Biology
In-depth discussion of problems and methods in computational molecular biology. Each year three topics will be covered and those will change yearly. Prerequisite: Biol 5495 or instructor’s consent.
Same as L41 Biol 5497
Credit 2 units.

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. 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 multiple alignment. 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, graduate standing, or permission of instructor.
Credit 3 units. EN: TU

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.