**Undergraduate Courses**

[**CSE 110: Computer Science Principles**](http://courses.cs.washington.edu/courses/cse120/) Introduces fundamental concepts of computer science and computational thinking. Includes logical reasoning, problem solving, data representation, abstraction, the creation of “digital artifacts” such as Web pages and programs, managing complexity, operation of computers and networks, effective Web searching, ethical, legal and social aspects of information technology.

[**CSE 121: Introduction to Computer Programming I**](https://courses.cs.washington.edu/courses/cse121/) Introduction to computer programming for students without previous programming experience. Students write programs to express algorithmic thinking and solve computational problems motivated by modern societal and scientific needs. Includes procedural programming constructs (methods), control structures (loops, conditionals), and standard data types, including arrays. Cannot be taken for credit if credit received for CSE 142. Recommended: completion of Paul G. Allen School's Guided Self-Placement.

[**CSE 122: Introduction to Computer Programming II**](https://courses.cs.washington.edu/courses/cse122/) Computer programming for students with some previous programming experience. Emphasizes program design, style, and decomposition. Uses data structures (e.g., lists, dictionaries, sets) to solve computational problems motivated by modern societal and scientific needs. Introduces data abstraction and interface versus implementation. Cannot be taken for credit if credit received for CSE 143. Recommended: CSE 121 or completion of Paul G. Allen School's Guided Self-Placement.

[**CSE 123: Introduction to Computer Programming III**](https://courses.cs.washington.edu/courses/cse123/) Computer programming for students with significant previous programming experience. Emphasizes implementation and run-time analysis of data structures and algorithms using techniques including linked references, recursion, and object-oriented inheritance to solve computational problems motivated by modern societal and scientific needs. Recommended: CSE 122 or completion of Paul G. Allen School's Guided Self-Placement.

[**CSE 131: Science and Art of Digital Photography**](http://courses.cs.washington.edu/courses/cse131/) Covers the fundamentals of digital photography, including computational imaging; the elements of photographic composition and design; and the future of internet-enabled photography.

[**CSE 142: Computer Programming I**](http://courses.cs.washington.edu/courses/cse142/) Basic programming-in-the-small abilities and concepts including procedural programming (methods, parameters, return, values), basic control structures (sequence, if/else, for loop, while loop), file processing, arrays, and an introduction to defining objects. Intended for students without prior programming experience.

[**CSE 143: Computer Programming II**](http://courses.cs.washington.edu/courses/cse143/) Continuation of CSE 142. Concepts of data abstraction and encapsulation including stacks, queues, linked lists, binary trees, recursion, instruction to complexity and use of predefined collection classes. Prerequisite: CSE 142.

[**CSE 154: Web Programming**](http://courses.cs.washington.edu/courses/cse154/) Covers languages, tools, and techniques for developing interactive and dynamic web pages. Topics include page styling, design, and layout; client and server side scripting; web security; and interacting with data sources such as databases. Prerequisite: a minimum grade of 2.0 in either CSE 122, CSE 123, CSE 142, CSE 143, CSE 160, or CSE 163.

[**CSE 160: Data Programming**](https://courses.cs.washington.edu/courses/cse160/) Introduction to computer programming. Assignments solve real data manipulation tasks from science, engineering, business, and the humanities. Concepts of computational thinking, problem-solving, data analysis, Python programming, control and data abstraction, file processing, and data visualization. Intended for students without prior programming experience. Cannot be taken for credit if credit received for either CSE 123 or CSE 143. (previously offered as CSE 140)

[**CSE 163: Intermediate Data Programming**](http://courses.cs.washington.edu/courses/cse163/) Intermediate data programming. Topics include writing programs that manipulate different types of data; leveraging the growing ecosystem of tools and libraries for data programming; writing programs that are both efficient and elegant; and writing medium-scale programs (100 to 200 lines). Prerequisite: either CSE 122, CSE 123, CSE 142, CSE 143, or CSE 160.

[**CSE 180: Introduction to Data Science**](http://courses.cs.washington.edu/courses/cse180/)Survey course introducing the essential elements of data science: data collection, management, curation, and cleaning; summarizing and visualizing data; basic ideas of statistical inference, machine learning. Students will gain hands-on experience through computing labs.

[**CSE 190b: Direct Admission Seminar**](http://courses.cs.washington.edu/courses/cse190b/) Seminar for freshmen admitted to CSE through Direct Freshman Admission.

[**CSE 311: Foundations Of Computing I**](http://courses.cs.washington.edu/courses/cse311/) Examines fundamentals of logic, set theory, induction, and algebraic structures with applications to computing; finite state machines; and limits of computability. Prerequisite: a minimum grade of 2.0 in either CSE 123 or CSE 143; and MATH 126 or MATH 136.

[**CSE 312: Foundations Of Computing Ii**](http://courses.cs.washington.edu/courses/cse312/) Examines fundamentals of enumeration and discrete probability; applications of randomness to computing; polynomial-time versus NP; and NP-completeness. Prerequisite: CSE 311.

[**CSE 331: Software Design And Implementation**](http://courses.cs.washington.edu/courses/cse331/) Explores concepts and techniques for design and construction of reliable and maintainable software systems in modern high-level languages: specifications; program structure and design; program-correctness approaches, including testing; and event-driven programming (e.g., graphical user interface). Prerequisite: a minimum grade of 2.0 in either CSE 123 or CSE 143.

[**CSE 332: Data Structures and Parallelism**](http://courses.cs.washington.edu/courses/cse332/) Covers abstract data types and structures including dictionaries, balanced trees, hash tables, priority queues, and graphs; sorting; asymptotic analysis; fundamental graph algorithms including graph search, shortest path, and minimum spanning trees; multithreading and parallel algorithms; P and NP complexity classes. No credit if CSE 373 has been taken. Prerequisite: CSE 311.

[**CSE 333: Systems Programming**](http://courses.cs.washington.edu/courses/cse333/) Includes substantial programming experience in languages that expose machine characteristics and low-level data representation (e.g., C and C++); explicit memory management; modern libraries and language features; interacting with operating-system services; introduction to concurrent programming. Prerequisite: CSE 351.

[**CSE 340: Interaction Programming**](https://courses.cs.washington.edu/courses/cse340/) User interfaces for computing systems, including principles and implementation techniques. Covers key topics and programming paradigms for interactive systems, such as event handling; graphical layout, design, and widgets; undo; accessibility; and context awareness. Provides experience with modern application domains and frameworks (e.g., mobile applications). Prerequisite: CSE 123 or CSE 143.

[**CSE 341: Programming Languages**](http://courses.cs.washington.edu/courses/cse341/) Basic concepts of programming languages, including abstraction mechanisms, static and dynamic typing, scope, syntax vs. semantics, first-class function closures, and algebraic datatypes. Detailed study of functional programming and other paradigms. Cannot be taken for credit if credit received for CSE 413. Prerequisite: CSE 123 or CSE 143.

[**CSE 344: Introduction To Data Management**](http://courses.cs.washington.edu/courses/cse344/) Introduces database management systems and writing applications that use such systems; data models (e.g., relational, semi-structured), query languages (e.g., SQL, XQuery), language bindings, conceptual modeling, transactions, security, database tuning, data warehousing, parallelism, and web-data management. Prerequisite: CSE 311.

[**CSE 351: The Hardware/Software Interface**](http://courses.cs.washington.edu/courses/cse351/) Examines key computational abstraction levels below modern high-level languages; number representation, assembly language, introduction to C, memory management, the operating-system process model, high-level machine architecture including the memory hierarchy, and how high-level languages are implemented. Prerequisite: CSE 123 or CSE 143.

[**CSE 369: Introduction to Digital Design**](https://courses.cs.washington.edu/courses/cse369/) Introduces the implementation, specification, and simulation of digital logic. Boolean algebra; combinational circuits including arithmetic circuits and regular structures; sequential circuits including finite-state-machines; and use of field-programmable gate arrays (FPGAs). Emphasizes simulation, high-level specification, and automatic synthesis techniques. Cannot be taken for credit if credit received for E E 271. Prerequisite: CSE 311.

[**CSE 371: Design of Digital Circuits and Systems**](https://www.ee.washington.edu/class/index.html)Provides a theoretical background in, and practical experience with, tools, and techniques for modeling complex digital systems with the Verilog hardware description language, maintaining signal integrity, managing power consumption, and ensuring robust intra- and inter-system communication. Prerequisite: either E E 205 or E E 215; either E E 271 or CSE 369. Offered: jointly with E E 371.

[**CSE 373: Data Structures And Algorithms**](http://courses.cs.washington.edu/courses/cse373/) Fundamental algorithms and data structures for implementation. Techniques for solving problems by programming. Linked lists, stacks, queues, directed graphs. Trees: representations, traversals. Searching (hashing, binary search trees, multiway trees). Garbage collection, memory management. Internal and external sorting. Intended for non-majors. Cannot be taken for credit if credit received for CSE 332. Prerequisite: CSE 123 or CSE 143.

[**CSE 374: Intermediate Programming Concepts And Tools**](http://courses.cs.washington.edu/courses/cse374/) Covers key software development concepts and tools not in introductory courses. Concepts of lower-level programming (C/C++) and explicit memory management; techniques and tools for individual and group software development; design, implementation, and testing strategies. Intended for non-majors. Cannot be taken for credit if credit received for CSE 333. Prerequisite: CSE 123 or CSE 143.

[**CSE 390t: Transfer Admit Seminar**](http://courses.cs.washington.edu/courses/cse390t/) Covers critical differences between UW CSE life and other schools based on previous transfer students' experiences. Topics will include significant differences between lecture and homework styles at UW, academic planning (graduating on time, class planning), and preparing for internships/industry. Also covers fundamentals to be successful in CSE 311 while juggling an exceptionally heavy course load.

[**CSE 390z: Mathematics for Computation Workshop**](https://courses.cs.washington.edu/courses/cse390z/) CSE 390Z is a workshop designed to provide academic support to students enrolled concurrently CSE 311. 311 concepts are at the core of computer science, but they are different from what many students have experienced in CS courses before. During each 2-hour workshop section for CSE 390Z, students will reinforce concepts through collaborative problem solving, practice study skills and effective learning habits, and build community for peer support. CSE 390Z is not a part of CSE 311 and is a separate course. All students enrolled in CSE 311 are welcome to register for this class.

[**CSE 391: System and Software Tools**](http://courses.cs.washington.edu/courses/cse391/) Introduction to tools commonly used in software development. Topics include using a command-line interface, writing scripts for file and string manipulation, managing user permissions, manipulating text with regular expressions, using build-management tools, and using version-control systems. Prerequisite: either CSE 122, CSE 123, or CSE 143. Credit/no-credit only.

[**CSE 399: CSE Foreign Study**](http://www.washington.edu/students/crscat/cse.html#cse399) Upper division computer science or computer engineering course, taken through an approved study abroad program, which there are no direct University of Washington equivalents. Credit/no credit only.

[**CSE 401: Introduction To Compiler Construction**](http://courses.cs.washington.edu/courses/cse401/) Fundamentals of compilers and interpreters; symbol tables; lexical analysis, syntax analysis, semantic analysis, code generation, and optimizations for general purpose programming languages. No credit to students who have taken CSE 413. Prerequisite: CSE 332; CSE 351.

[**CSE 402: Design and Implementation of Domain-Specific Languages**](http://courses.cs.washington.edu/courses/cse402/)Design and implementation of domain-specific languages. Creation of new programming abstractions, formal and informal language specification techniques, implementation strategies to support language analysis and execution on traditional and non-traditional computing platforms. Selection and use of appropriate software tools and development environments to build novel DSLs. Prerequisite: CSE 332 and CSE 351.

[**CSE 403: Software Engineering**](http://courses.cs.washington.edu/courses/cse403/) Fundamentals of software engineering using a group project as the basic vehicle. Topics covered include the software crisis, managing complexity, requirements specification, architectural and detailed design, testing and analysis, software process, and tools and environments. Prerequisite: CSE 331; CSE 332.

[**CSE 410: Computer Systems**](http://courses.cs.washington.edu/courses/cse410/) Structure and components of hardware and software systems. Machine organization, including central processor and input-output architectures; assembly language programming; operating systems, including process, storage, and file management. Intended for non-majors. No credit to students who have completed CSE 351 or CSE 451. Prerequisite: CSE 373.

[**CSE 412: Introduction to Data Visualization**](http://courses.cs.washington.edu/courses/cse412/) Introduction to data visualization design and use for both data exploration and explanation. Methods for creating effective visualizations using principles from graphic design, psychology, and statistics. Topics include data models, visual encoding methods, data preparation, exploratory analysis, uncertainty, cartography, interaction techniques, visual perception, and evaluation methods. Cannot be taken for credit if credit received for CSE 442. Prerequisite: either CSE 123, CSE 143, or CSE 163.

[**CSE 413: Programming Languages & Their Implementation**](http://courses.cs.washington.edu/courses/cse413/) Basic concepts and implementation strategies for modern functional and object-oriented programming languages such as Scheme and Java. Intended for non-majors. No credit to students who have completed CSE 341 or CSE 401. Prerequisite: CSE 373.

[**CSE 414: Introduction to Database Systems**](http://courses.cs.washington.edu/courses/cse414/) Introduces database management systems and writing applications that use such systems; data models, query languages, transactions, database tuning, data warehousing, and parallelism. Intended for non-majors. Cannot be taken for credit if credit received for CSE 344. Prerequisite: a minimum grade of 2.5 in either CSE 123, CSE 143, or CSE 163.

[**CSE 415: Introduction To Artificial Intelligence**](http://courses.cs.washington.edu/courses/cse415/) Principles and programming techniques of artificial intelligence: Python, symbol manipulation, knowledge representation, logical and probabilistic reasoning, learning, language understanding, vision, expert systems, and social issues. Intended for non-majors. Not open for credit to students who have completed CSE 473. Prerequisite: CSE 373.

[**CSE 416: Introduction to Machine Learning**](http://courses.cs.washington.edu/courses/cse416/)Provides practical introduction to machine learning. Modules include regression, classification, clustering, retrieval, recommender systems, and deep learning, with a focus on an intuitive understanding grounded in real-world applications. Intelligent applications are designed and used to make predictions on large, complex datasets. Prerequisite: either CSE 123, CSE 143, CSE 160, or CSE 163; and either STAT 311, STAT 390, STAT 391, IND E 315, MATH 394/STAT 394, STAT 395/MATH 395, or Q SCI 381. Offered: jointly with STAT 416.

[**CSE 417: Algorithms And Computational Complexity**](http://courses.cs.washington.edu/courses/cse417/) Design and analysis of algorithms and data structures. Efficient algorithms for manipulating graphs and strings. Fast Fourier Transform. Models of computation, including Turing machines. Time and space complexity. NP-complete problems and undecidable problems. Intended for non-majors. Prerequisite: CSE 373.

[**CSE 421: Introduction To Algorithms**](http://courses.cs.washington.edu/courses/cse421/) Techniques for design of efficient algorithms. Methods for showing lower bounds on computational complexity. Particular algorithms for sorting, searching, set manipulation, arithmetic, graph problems, pattern matching. Prerequisite: CSE 312; CSE 332.

[**CSE 422: Toolkit for Modern Algorithms**](https://courses.cs.washington.edu/courses/cse422/) A rigorous introduction to the principles of modern algorithm design, with a particular focus on the analysis of large, noisy data sets, and the algorithmic principles underlying modern statistics and machine learning. Students reason about and implement contemporary algorithms, and analyze their behavior on real-world data sets. Prerequisite: CSE 312; CSE 332; and MATH 208.

[**CSE 426: Cryptography**](http://courses.cs.washington.edu/courses/cse426/)Rigorous introduction to modern cryptography. Design and application of important cryptographic objects, such as encryption, message authentication, and zero-knowledge proofs. Students learn to formalize security goals, design schemes for achieving these goals, and study security attacks or security proofs that establish the security or insecurity of schemes. Prerequisite: CSE 312.

[**CSE 427: Computational Biology**](http://courses.cs.washington.edu/courses/cse427/) Algorithmic and analytic techniques underlying analysis of large-scale biological data sets such as DNA, RNA, and protein sequences or structures, expression and proteomic profiling. Hands-on experience with databases, analysis tools, and genome markers. Applications such as sequence alignment, BLAST, phylogenetics, and Markov models. Prerequisite: CSE 312; CSE 332.

[**CSE 428: Computational Biology Capstone**](http://courses.cs.washington.edu/courses/cse428/) Designs and implements a software tool or software analysis for an important problem in computational molecular biology. Prerequisite: CSE 312; CSE 331; CSE 332.

[**CSE 431: Introduction To Theory Of Computation**](http://courses.cs.washington.edu/courses/cse431/) Models of computation, computable and noncomputable functions, space and time complexity, tractable and intractable functions. Prerequisite: CSE 312.

[**CSE 440: Intro To HCI: User Interface Design, Prototype, Evaluation**](http://courses.cs.washington.edu/courses/cse440/) Human-Computer Interaction (HCI) theory and techniques. Methods for designing, prototyping, and evaluating user interfaces to computing applications. Human capabilities, interface technology, interface design methods, and interface evaluation tools and techniques. Prerequisite: CSE 332.

[**CSE 441: Advanced Hci: User Interface Dsgn, Prototype, Eval**](http://courses.cs.washington.edu/courses/cse441/) Human-Computer Interaction (HCI) theory and techniques. Advanced methods for designing, prototyping, and evaluating user interfaces to computing applications. Novel interface technology, advanced interface design methods, and prototyping tools. Prerequisite: CSE 440.

[**CSE 442: Data Visualization**](http://courses.cs.washington.edu/courses/cse442/) Techniques for creating effective visualizations of data based on principles from graphic design, perceptual psychology, and statistics. Topics include visual encoding models, exploratory data analysis, visualization software, interaction techniques, graphical perception, color, animation, high-dimensional data, cartography, network visualization, and text visualization. Prerequisite: CSE 332.

[**CSE 444: Database Systems Internals**](http://courses.cs.washington.edu/courses/cse444/)

The relational data model and the SQL query language. Conceptual modeling: entity/relationships, normal forms. XML, XPath, and XQuery. Transactions: recovery and concurrency control. Implementation of a database system. A medium sized project using a rational database backend. Prerequisite: CSE 332; and either CSE 344 or CSE 414.

[**CSE 446: Machine Learning**](http://courses.cs.washington.edu/courses/cse446/) Design of efficient algorithms that learn from data. Representative topics include supervised learning, unsupervised learning, regression and classification, deep learning, kernel methods, and optimization. Emphasis on algorithmic principles and how to use these tools in practice. Prerequisite: CSE 332; MATH 208 or MATH 136; and either STAT 390, STAT 391, or CSE 312.

[**CSE 447: Natural Language Processing**](http://courses.cs.washington.edu/courses/cse447/)Methods for designing systems that usefully and/or intelligently process natural language text data. Language models, text categorization, syntactic and semantic analysis, machine translation. This course emphasizes algorithms and data-driven methods. Prerequisite: CSE 312 and CSE 332; recommended: MATH 208 ; CSE 446 is recommended before or concurrently.

[**CSE 450: Animation Production Seminar**](http://courses.cs.washington.edu/courses/cse450/) Open to all students who have an interest in digital animation. Reviews and analyzes films, animated feature films, and television commercials. Emphasizes the technical and aesthetic basics of animation production in industry studio environments.

[**CSE 451: Introduction To Operating Systems**](http://courses.cs.washington.edu/courses/cse451/) Principles of operating systems. Process management, memory management, auxiliary storage management, resource allocation. No credit to students who have completed CSE 410. Prerequisite: CSE 351; CSE 332; CSE 333.

[**CSE 452: Introduction to Distributed Systems**](http://courses.cs.washington.edu/courses/cse452/) Covers abstractions and implementation techniques in the construction of distributed systems, including cloud computing, distributed storage systems, and distributed caches. Prerequisite: CSE 332 and CSE 333; recommended: CSE 451

[**CSE 453: Datacenter Systems**](https://courses.cs.washington.edu/courses/cse453/) Datacenter server, network, storage, security, and energy management technologies. Virtualization, I/O architectures, resource isolation and provisioning, disaggregation, quality of service, and tail latency. Prerequisite: CSE 332 and CSE 333; recommended: CSE 451 or CSE 452.

[**CSE 455: Computer Vision**](http://courses.cs.washington.edu/courses/cse455/) Introduction to image analysis and interpreting the 3D world from image data. Topics may include segmentation, motion estimation, image mosaics, 3D-shape reconstruction, object recognition, and image retrieval. Prerequisite: CSE 333; CSE 332; recommended: MATH 208; STAT 391.

[**CSE 456: Story Design For Computer Animation**](http://courses.cs.washington.edu/courses/cse456/) Animation principles and production for story development and design. Design, development, and production of several storyreels, which are a tool for the pre-production of animated features and shorts. Student use authoring tools to present finished work.

[**CSE 457: Computer Graphics**](http://courses.cs.washington.edu/courses/cse457/) Introduction to computer image synthesis, modeling, and animation. Topics may include visual perception, displays and framebuffers, image processing, affine and projective transformations, hierarchical modeling, hidden surface elimination, shading, ray-tracing, anti-aliasing, texture mapping, curves, surfaces, particle systems, dynamics, character animation, and animation principles. Prerequisite: CSE 333; CSE 332; recommended: CSE 312; MATH 208.

[**CSE 458: Computer Animation**](http://courses.cs.washington.edu/courses/cse458/) Introduction to basic principles of computer generated animation. Focus on the modeling and lighting of animated characters. Students from art, CSE, and music team up on projects to be built on commercially-available modeling and lighting packages. Prerequisite: CSE 457; recommended: instructor permission.

[**CSE 459: Pre-production For Collaborative Animation**](http://courses.cs.washington.edu/courses/cse459/) Pre-production of collaboratively designed animated shorts. In-depth analysis of classical and computer generated works. Character design and pre-planning, model sheets, character rigging, storyreel and animatics, character motion, design for multiple characters, and principles of animation as applied to character motion and effects. Prerequisite: CSE 458.

[**CSE 460: Animation Capstone**](http://courses.cs.washington.edu/courses/cse460/) Apply the knowledge gained in previous animation courses to produce a short animated film. Topics include scene planning, digital cinematography, creature and hard surface modeling, animatics and basics of character animation, and rendering techniques. Prerequisite: CSE 458, CSE 459.

[**CSE 461: Introduction To Computer Communication Networks**](http://courses.cs.washington.edu/courses/cse461/) Computer network architectures, protocol layers, network programming. Transmission media, encoding systems, switching, multiple access arbitration. Network routing, congestion control, flow control. Transport protocols, real-time, multicast, network security. Prerequisite: CSE 332; CSE 333.

[**CSE 462: Wireless Communication**](http://courses.cs.washington.edu/courses/cse462) Fundamentals of wireless communication. Reliable communication over noisy, unreliable channels. Signal to noise ratio, frequency domain analysis, bandwidth, capacity of noisy communication channels, modulation, channel coding, error detection, error correction. Connection between machine learning and communication such as decoding as inference and learning as compression.

[**CSE 464: Advanced Topics In Digital Animation**](http://courses.cs.washington.edu/courses/cse464/) Students design individual animated works for professional quality demo reels. 2- and 3-D animatics, special effects design, advanced character animation techniques, 3-D paint techniques and integration, short design, sequence planning, non-photorealistic rendering options, interactive animation for pre-planning, and advanced production techniques and strategies. Recommended: CSE 458.

[**CSE 467: Advanced Digital Design**](http://courses.cs.washington.edu/courses/cse467/) Advanced techniques in the design of digital systems. Hardware description languages, combinational and sequential logic synthesis and optimization methods, partitioning, mapping to regular structures. Emphasis on reconfigurable logic as an implementation medium. Memory system design. Digital communication including serial/parallel and synchronous/asynchronous methods. Prerequisite: CSE 352; CSE 332.

[**CSE 468: Very Large Scale Integration**](http://courses.cs.washington.edu/courses/cse468/) Introduction to CMOS technology and circuit design; implementation of combinational and sequential logic; VLSI design methodologies; CAD tools for layout, simulation, and validation. Students design a VLSI chip using modern CAD tools. Prerequisite: either CSE 352 or CSE 370.

[**CSE 469: Computer Architecture I**](https://www.ee.washington.edu/class/index.html) Introduction to computer architecture. Assembly and machine language, microprocessor organization including control and datapath. Computer arithmetic. Memory systems and caching. Performance modeling of microprocessors. Prerequisite: either E E 271 or CSE 369; CSE 143 Offered: jointly with E E 469.

[**CSE 470: Computer Architecture II**](http://courses.cs.washington.edu/courses/cse470/) Advanced computer architecture. Performance evaluation and energy efficiency. Instruction set architectures. Instruction-level parallelism. Modern microprocessor micro-architecture. Thread-level parallelism. Cache coherency and memory consistency in shared-memory multiprocessors. Memory hierarchy. GPU architecture. Warehouse-scale computing. Trends in computer design. Prerequisite: CSE 351; either CSE 469 or E E 469. Offered: jointly with E E 470. (previously offered as CSE 471)

[**CSE 472: Introduction To Computational Linguistics**](http://www.washington.edu/students/crscat/cse.html#cse472) Introduction to computational approaches to modeling language, for linguistic research and practical applications, including analyses at different levels of linguistic structure and symbolic as well as statistical approaches. Prerequisite: either LING 200 or LING 400; either LING 461 or CSE 311. Offered: jointly with LING 472.

[**CSE 473: Introduction To Artificial Intelligence**](http://courses.cs.washington.edu/courses/cse473/) Principal ideas and developments in artificial intelligence: Problem solving and search, game playing, knowledge representation and reasoning, uncertainty, machine learning, natural language processing. Not open for credit to students who have completed CSE 415. Prerequisite: CSE 312; CSE 332.

[**CSE 474: Introduction to Embedded Systems**](http://courses.cs.washington.edu/courses/cse474/) Introduces the specification, design, development, and test of real time embedded system software. Use of a modern embedded microcomputer or microcontroller as a target environment for a series of laboratory projects and a comprehensive final project. Prerequisite: CSE 143. Offered: jointly with E E 474;

[**CSE 475: Embedded Systems Capstone**](http://courses.cs.washington.edu/courses/cse475/) Capstone design experience. Prototype a substantial project mixing hardware, software, and communications. Focuses on embedded processors, programmable logic devices, and emerging platforms for the development of digital systems. Provides a comprehensive experience in specification, design, and management of contemporary embedded systems. Prerequisite: either E E 271 or CSE 369; either CSE 466, E E 472, or CSE 474/E E 474. Offered: jointly with E E 475;

[**CSE 478: Autonomous Robotics**](https://courses.cs.washington.edu/courses/cse478/)Theory and application of algorithms and probabilistic techniques for autonomous robotics. Covers topics related to state estimation (Bayes filtering, probabilistic motion and sensor models), planning/control (search based planners, lattice based planners, trajectory following techniques), and perception and learning (object detection, learning from demonstrations etc.). Prerequisite: CSE 332; recommended: CSE 312; MATH 208.

[**CSE 480: Computer Ethics Seminar**](https://courses.cs.washington.edu/courses/cse480/) Reading and discussion seminar on the ethical issues surrounding sociotechnical systems and their impacts on society. Includes historical perspectives and current topics. Relates computer technology to ethical questions of power, politics, equity, privacy, autonomy, and more. Credit/no-credit only.

[**CSE 481: Capstone Software Design**](https://www.cs.washington.edu/students/ugrad/capstone) Students work in teams to design and implement a software project involving multiple areas of the CSE curriculum. Emphasis is placed on the development process itself, rather than on the product. Prerequisite: Instructor permission or CSE 332; CSE 351; and either CSE 331 or CSE 352; recommended that students have completed 300-level courses and several 400-level CSE courses.

**Links to individual capstone course web pages listed below.**

[**CSE 481a: Capstone Software: Operating Systems**](http://courses.cs.washington.edu/courses/cse481a/) Students work in substantial teams to design, implement, and release a software project involving multiple areas of the CSE curriculum. Emphasis is placed on the development process itself, rather than on the product. Teams are expected to develop a work plan, and to track and document their progress against it.

[**CSE 481c: Robotics Capstone**](http://courses.cs.washington.edu/courses/cse481c/) Students work in teams to design and implement algorithms for robotic perception and control.

[**CSE 481d: Capstone Software: Games**](http://courses.cs.washington.edu/courses/cse481d/) Students work in substantial teams to design, implement, and release a software project involving multiple areas of the CSE curriculum. Emphasis is placed on the development process itself, rather than on the product. Teams are expected to develop a work plan, and to track and document their progress against it.

[**CSE 481e: Capstone Software: UrbanSim**](http://courses.cs.washington.edu/courses/cse481e/) UrbanSim Capstone

[**CSE 481g: Capstone Software: Distributed Systems**](http://courses.cs.washington.edu/courses/cse481g/) Distributed systems capstone.

[**CSE 481h: Capstone Software: Accessibility**](http://courses.cs.washington.edu/courses/cse481h/) Accessibility capstone.

[**CSE 481i: Sound Capstone**](http://courses.cs.washington.edu/courses/cse481i/) Sound Capstone

[**CSE 481k: Capstone Software: Designing Tech. for Resource-Constrained Envs.**](http://courses.cs.washington.edu/courses/cse481k/) Students form interdisciplinary project groups to scope and design projects for resource-constrained environments. The emphasis is on group work leading to the creation of testable realizations and completion of initial evaluations of the software and hardware artifacts produced. Students work in inter-disciplinary groups with a faculty or graduate student manager. Groups document their work in the form of posters, verbal presentations, videos, and written reports.

[**CSE 481m: Home Networking Capstone**](http://courses.cs.washington.edu/courses/cse481m/) Home Networking Capstone

[**CSE 481O: Capstone Software - Kinect**](http://courses.cs.washington.edu/courses/cse481o/) Students work in teams to design and implement a software project that makes use of RGB-D sensors (e.g. Microsoft Kinect, ASUS Xtion Pro Live)

[**CSE 481s: Computer Security Capstone**](https://courses.cs.washington.edu/courses/cse481s/) Student teams will be tasked with creating a computer security themed product. The work will progress from product conception to requirements to design to implementation to evaluation. Along the way, students will incorporate key computer security tools and practices, including threat modeling, penetration testing, and bug fixing. Examples include password managers, censorship resistance systems, and mobile payment systems.

[**CSE 481S: Computer Security Capstone**](http://courses.cs.washington.edu/courses/cse481s/) Student teams will be tasked with creating a computer security themed product. The work will progress from product conception to requirements to design to implementation to evaluation. Along the way, students will incorporate key computer security tools and practices, including threat modeling, penetration testing, and bug fixing.

[**CSE 481v: Virtual and Augmented Reality Capstone**](https://courses.cs.washington.edu/courses/cse481v/) Learn a ton about Virtual and Augmented Reality, get familiar with the latest technology and software, and build an app in 10 weeks.

[**CSE 484: Computer Security**](http://courses.cs.washington.edu/courses/cse484/) Foundations of modern computer security, including software security, operating system security, network security, applied cryptography, human factors, authentication, anonymity, and web security. Prerequisite: CSE 332; CSE 351.

[**CSE 486: Introduction To Synthetic Biology**](http://courses.cs.washington.edu/courses/cse486/) Studies mathematical modeling of transcription, translation, regulation, and metabolism in cell; computer aided design methods for synthetic biology; implementation of information processing, Boolean logic and feedback control laws with genetic regulatory networks; modularity, impedance matching and isolation in biochemical circuits; and parameter estimation methods. Prerequisite: either MATH 136 or MATH 307, AMATH 351, or CSE 311 and MATH 208 or AMATH 352. Offered: jointly with BIOEN 423/E E 423.

[**CSE 487: Advanced Systems And Synthetic Biology**](http://courses.cs.washington.edu/courses/cse487/) Introduces advanced topics in systems and synthetic biology. Topics include advanced mathematical modeling; computational standards; computer algorithms for computational analysis; and metabolic flux analysis, and protein signaling pathways and engineering. Prerequisite: either BIOEN 401, BIOEN 423,E E 423, or CSE 486. Offered: jointly with BIOEN 424/E E 424; W.

[**CSE 488: Laboratory Methods In Synthetic Biology**](http://courses.cs.washington.edu/courses/cse488/) Designs and builds transgenic bacterial using promoters and genes taken from a variety of organisms. Uses construction techniques including recombination, gene synthesis, and gene extraction. Evaluates designs using sequencing, fluorescence assays, enzyme activity assays, and single cell studies using time-lapse microscopy. Prerequisite: either BIOEN 423, E E 423, or CSE 486; either CHEM 142, CHEM 144, or CHEM 145. Offered: jointly with BIOEN 425/E E 425.

[**CSE 490A: Entrepreneurship**](https://courses.cs.washington.edu/courses/cse599a1/)Company-Building from Formation to Successful Exit. This course is about entrepreneurship and specifically about starting, growing, managing, leading, and ultimately exiting a new venture. Offered jointly with CSE 599 A1, CSE P 590, ENTRE 532.

[**CSE 490as: Animation Production Seminar**](http://courses.cs.washington.edu/courses/cse490as/) Animation Production Seminar

[**CSE 490d: Hardware Capstone Project Exploration**](http://courses.cs.washington.edu/courses/cse490d/) Hardware Capstone Project Exploration

[**CSE 490dv: Story Design for Computer Animation**](http://courses.cs.washington.edu/courses/cse456/) Animation principles and production for story development and design. Design, development, and production of several storyreels, which are a tool for the pre-production of animated features and shorts. Student use authoring tools to present finished work.

[**CSE 490i: Neurobotics**](http://courses.cs.washington.edu/courses/cse490i/) The field of Neurobotics lies at the intersection of robotics and medicine. It aims to build a robot-human closed loop system to alter the neural control of movement as a way to rehabilitate, assist, and enhance human motor control and learning capabilities. Typically, the primary target population is individuals with strokes, spinal cord injuries, traumatic brain injuries, and other injuries that inhibit daily activities. However, it could also target sports medicine, military, and entertainment applications. This course is an introductory design course in Neurobotics focusing on learning about human neural control of movement, using physiological signals as inputs, and controlling a mechanical device. Students will learn simple control laws, hands on experience and programming in controlling robots, and applying knowledge of human movements to move the robot. There is a design project competition at the end of quarter.

[**CSE 490j: Special Topics in Computer Animation**](http://courses.cs.washington.edu/courses/cse490j/) Special Topics in Computer Animation

[**CSE 490jl: User Interface Design, Prototyping, and Evaluation**](http://courses.cs.washington.edu/courses/cse490f/) User Interface Design, Prototyping, and Evaluation

[**CSE 490st: Educational Software Capstone**](http://courses.cs.washington.edu/courses/cse490st/) Software to support learning comprises a wide variety of styles and scales of programming. From games for children to large online learning systems, educational software not only touches on many areas of computer science but depends also on psychology, communication, design, and the fields of study the software is to support. Developing software for learning can be both highly exciting and very challenging. This capstone course will blend a study of key aspects of educational software with group projects that allow students to explore the issues in a hands-on manner. Key issues include: identifying learning objectives, design of simulations and construction sets, managing the learning curve for the software itself, interface design, supporting motivation, supporting collaborative learning and communication, educational assessment, connecting to standards, integration with environments and platforms, evaluation methods, and deployment approaches.

[**CSE 490v: Virtual Reality Systems**](http://courses.cs.washington.edu/courses/cse490v/) Modern virtual reality systems draw on the latest advances in optical fabrication, embedded computing, motion tracking, and real-time rendering. In this hands-on course, students will foster similar cross-disciplinary knowledge to build a fully functional head-mounted display. This overarching project spans hardware (optics, displays, electronics, and microcontrollers) and software (JavaScript, WebGL, and GLSL). Each assignment will build toward this larger goal. For example, in one assignment, students will learn to use an inertial measurement unit (IMU) to track the position of the headset. In another assignment, students will apply real-time computer graphics methods to correct lens distortions. Lectures will complement these engineering projects, diving into the history of AR/VR and relevant topics in computer graphics, signal processing, and human perception. Guest speakers will participate from leading AR/VR companies and academic institutions. Prerequisites: MATH 208; CSE 333. Recommended: CSE 455 or CSE 457). Familiarity with JavaScript will be helpful, but is not required.

[**CSE 491: Data Science and Society Seminar**](http://courses.cs.washington.edu/courses/cse491/) Current topics related to the societal implications of data science. Topic selection will vary from quarter to quarter and may include data privacy and security, data anonymization, hypothesis-testing on a shared database, impact of data science-based decisions on society. Includes both guest speakers and case-study or article-based discussions. Credit/no-credit only.

[**CSE 492L: Alumni Career Experience Series**](http://courses.cs.washington.edu/courses/cse492l) The Allen School Alumni Experience Series, CSE 402L, is a one-credit (CR/NC) seminar series, targeted at undergraduates, that brings CSE alumni and friends to campus to describe how to be effective in a startup, small company, large company, or less common environment. Course requirements: Regular attendance, active participation, and some preparation on a wiki in advance of each course session. (Formerly called the Leadership Seminar Series.)

[**CSE 493: Advanced Special Topics in Computer Science and Engineering**](http://courses.cs.washington.edu/courses/cse493/) Advanced topics in computer science and engineering.

[**CSE 495: Project Practicum**](http://courses.cs.washington.edu/courses/cse495/) Available in special situations for computer science majors to compete, under instructor guidance, a substantial computing project that deepens one's knowledge and experience in the field. Projects may involve a group of students.

[**CSE 496: Honors Undergraduate Research**](http://courses.cs.washington.edu/courses/cse496/) Undergraduate research or design project carried out under the supervision of a faculty sponsor.

[**CSE 497: Undergraduate Research Seminar**](http://courses.cs.washington.edu/courses/cse497/) Students prepare and give a public talk on their faculty-sponsored research projects.

[**CSE 498: Senior Project**](http://courses.cs.washington.edu/courses/cse498/) A report (and perhaps demonstration) describing a development, survey, or small research project in computer science or an application to another field. Objectives are: (1) integrating material from several courses, (2) introducing the professional literature, (3) gaining experience in writing a technical document, and (4) showing evidence of independent work. Work normally extends over more than one quarter, for a maximum of 6 credits for 498; 9 credits are required for 498H.

[**CSE 499: Reading And Research**](http://www.washington.edu/students/crscat/cse.html#cse499) Available in special situations for advanced computer science majors to do reading and research in field, subject to approval of undergraduate adviser and CSE faculty member. Free elective, but does not replace core course or computer science elective. Credit/no credit only.

**Professional (Evening) Courses**

[**CSEP 501: Compiler Construction**](http://courses.cs.washington.edu/courses/csep501/) Principles and practice of building efficient implementations of modern programming languages. Lexical, syntactic, and semantic analysis of programs. Intermediate program representations. Intra- and interprocedural analysis and optimization. Run-time system techniques. Related programming environment facilities such as source-level debuggers and profilers. Prerequisite: CSE majors only.

[**CSEP 503: Principles Of Software Engineering**](http://courses.cs.washington.edu/courses/csep503/)

Study of major developments in software engineering over the past three decades. Topics may include design (information hiding, layering, open implementations), requirements specification (informal and formal approaches), quality assurance (testing, verification and analysis, inspections), reverse and re-engineering (tools, models, approaches). Prerequisite: CSE majors only.

[**CSEP 504: Advanced Topics In Software Systems**](http://courses.cs.washington.edu/courses/csep504/) Topics include software architecture, software tools, programming language analysis, type systems, formal reasoning, and other pertinent topics in software engineering and programming languages research.

[**CSEP 505: Programming Languages**](http://courses.cs.washington.edu/courses/csep505/) A study of non-imperative programming paradigms such as functional, object-oriented, logic, and constraint programming. Programming language semantics and type theory. Prerequisite: CSE majors only.

[**CSEP 506: Advanced Topics in Programming Languages**](http://courses.cs.washington.edu/courses/csep506/) May include functional, object-oriented, parallel, and logic programming languages; semantics for languages of these kinds; type declaration, inference, and checking (including polymorphic types); implementation issues, such as compilation, lazy evaluation, combinators, parallelism, various optimization techniques. Implementation project required.

[**CSEP 510: Human Computer Interaction**](http://courses.cs.washington.edu/courses/csep510/)

Topics in human-computer interaction, including tools and skills for user interface design, user interface software architecture, rapid prototyping and iterative design, safety and critical systems, evaluation techniques, and computer supported cooperative work. Prerequisite: CSE majors only.

[**CSEP 517: Natural Language Processing**](http://courses.cs.washington.edu/courses/csep517/) Overview of modern approaches for natural language processing. Topics include language models, text classification, tagging, parsing, machine translation, semantics, and discourse analysis.

[**CSEP 521: Applied Algorithms**](http://courses.cs.washington.edu/courses/csep521/) Principles of design of efficient algorithms with emphasis on algorithms with real world applications. Examples drawn from computational geometry, biology, scientific computation, image processing, combinatorial optimization, cryptography, and operations research. Prerequisite: CSE majors only.

[**CSEP 524: Parallel Computation**](http://courses.cs.washington.edu/courses/csep524/) Survey of parallel computing including the processing modes of pipelining, data parallelism, thread parallelism, and task parallelism; algorithmic implications of memory models; shared memory and message passing; hardware implementations; bandwidth and latency; synchronization, consistency, interprocessor communication; programming issues including implicit and explicit parallelism, locality, portability. CSE majors only.

[**CSEP 527: Computational Biology**](https://courses.cs.washington.edu/courses/csep527/) Introduction to the use of computational methods for understanding biological systems at the molecular level. Problem areas such as mapping and sequencing, sequence analysis, structure prediction, phylogenic inference, motif discovery, expression analysis, and regulatory analysis. Techniques such as dynamic programming, Markov models, MCMC, expectation-maximization, and local search.

[**CSEP 531: Computability And Complexity Theory**](http://courses.cs.washington.edu/courses/csep531/) Survey of the theory of computation including Turing Machines, Churche' s Thesis, computability, incompleteness, undecidability, complexity classes, problem reductions, Cook' s theorem, NP-completeness, randomized computation, cryptography, parallel computation, and space complexity. Some emphasis will be placed on historical and philosophical aspects of the theory of computation. Prerequisite: CSE PMP majors only.

[**CSEP 544: Database Management Systems**](http://courses.cs.washington.edu/courses/csep544/) Introduction to the principles of database management systems. Topics include database system architecture, data models, theory of database design, query optimization, concurrency control, crash recovery, and storage strategies. CSE majors only.

[**CSEP 545: Transaction Processing**](http://courses.cs.washington.edu/courses/csep545/)

Technology supporting reliable large-scale distributed computing, including transaction programming models, TP monitors, transactional communications, persistent queuing, software fault tolerance, concurrency control and recovery algorithms, distributed transactions, two-phase commit, data replication. Prerequisite: CSE majors only.

[**CSEP 546: Machine Learning**](http://courses.cs.washington.edu/courses/csep546/)

Methods for designing systems that learn from data and improve with experience. Supervised learning and predictive modeling; decision trees, rule induction, nearest neighbors, Bayesian methods, neural networks, support vector machines, and model ensembles. Unsupervised learning and clustering. Prerequisite: CSE PMP majors only.

[**CSEP 548: Computer Architecture**](http://courses.cs.washington.edu/courses/csep548/) Architecture of the single-chip microprocessor: instruction set design and processor implementation (pipelining, multiple issue, speculative execution). Memory hierarchy: on-chip and off-chip caches, TLBs and their management, virtual memory from the hardware viewpoint. I/O devices and control: buses, disks, and RAIDs. Prerequisite: CSE majors only.

[**CSEP 551: Computer Operating Systems**](http://courses.cs.washington.edu/courses/csep551/)

A study of developments in operating systems from the 196' s to the present. Topics include operating system structure, protection, virtual memory, communication mechanisms, concurrency, lightweight threads, object-oriented systems, distributed systems, and transaction support in operating systems. Prerequisite: CSE majors only.

[**CSEP 552: Distributed Systems**](http://courses.cs.washington.edu/courses/csep552/) Principles, techniques, and examples related to the design, implementation, and analysis of distributed computer systems. Prerequisite: CSE PMP majors only.

[**CSEP 557: Current Trends In Computer Graphics**](http://courses.cs.washington.edu/courses/csep557/) Introduction to computer image synthesis, modeling, and animation emphasizing the state-of-the-art algorithm applications. Topics may include visual perception, image processing, geometric transformations, hierarchical modeling, hidden-surface elimination, shading, ray-tracing, anti-aliasing, texture mapping, curves, surfaces, particle systems, dynamics, realistic character animation, and traditional animation principles. Prerequisite: CSE majors only.

[**CSEP 561: Network Systems**](http://courses.cs.washington.edu/courses/csep561/)

Current choices and challenges in network systems. Fundamental concepts combined with emphasis on evaluation of design/operations alternatives. Topics include alternative link, network, and transport-layer technologies, topologies, routing, congestion control multimedia, Ipv6, aTM v. IP, network management and policy issues. Prerequisite: CSE majors only.

[**CSEP 567: Design And Implementation Of Digital Systems**](http://courses.cs.washington.edu/courses/csep567/) Overview of current implementation technologies for digital systems including custom integrated circuits, field-programmable logic, and embedded processors. Systems components such as buses and communications structures, interfaces, memory architectures, embedded systems, and application-specific devices. Focus on the design of large systems using modern CAD tools. Prerequisite: CSE majors only.

[**CSEP 573: Applications Of Artificial Intelligence**](http://courses.cs.washington.edu/courses/csep573/) Introduction to the use of Artificial Intelligence tools and techniques in industrial and company settings. Topics include foundations (search, knowledge representation) and tools such as expert systems, natural language interfaces and machine learning techniques. Prerequisite: CSE majors only.

[**CSEP 576: Computer Vision**](http://courses.cs.washington.edu/courses/csep576/) Provides an overview of computer vision, emphasizing the middle ground between image processing and artificial intelligence. Image formation, pre-attentive image processing, boundary and region representations, and case studies of vision architectures.

[**CSEP 585: Design and Implementation of Digital Systems**](https://courses.cs.washington.edu/course_archives/www_archive/cse585/) Overview of current implementation technologies for digital systems including custom integrated circuits, field-programmable logic, and embedded processors. Systems components such as buses and communications structures, interfaces, memory architectures, embedded systems, and application-specific devices. Focus on the design of large systems using modern CAD tools.

[**CSEP 589: Software Entrepreneurship**](https://courses.cs.washington.edu/courses/csep589/) A case- and project-based course, focusing on starting a software or hardware company. Guest entrepreneurs, lawyers, and financiers discuss market identification and analysis, planning the business, financing, and typical operating and administrative problems. Offered: jointly with CSE P 589/ENTRE 532.

[**CSEP 590: Special Topics In Computer Science (PMP)**](http://courses.cs.washington.edu/courses/csep590/) Studies of emerging areas and specialized topics in computer science. Topics vary by quarter. For PMP students, repeatable with 20-credit limit.

[**CSEP 590: Entrepreneurship**](https://courses.cs.washington.edu/courses/cse599a1/) Company-Building from Formation to Successful Exit. This course is about entrepreneurship and specifically about starting, growing, managing, leading, and ultimately exiting a new venture. For PMP students, counts toward 20-credit CSE P 590 limit. Offered jointly with CSE 599A1, CSE P 590, ENTRE 532.

[**CSEP 595: Software Entrepreneurship**](http://courses.cs.washington.edu/courses/csep595/) Provides an overview of the major elements of entrepreneurial activity in software, including market identification and analysis, evaluation and planning of the business, financing, typical operating and administrative problems, and alternatives for growth or sale.

**Group Meetings**

[**CSE 590d: Computer-Based Learning Environments**](http://courses.cs.washington.edu/courses/cse590d/) Computer-Based Learning Environments

[**CSE 591a: Programming Systems Group Meeting**](http://courses.cs.washington.edu/courses/cse591a/) Programming Systems Group Meeting

[**CSE 591b: Brain-Computer Interfaces Group Meeting**](http://courses.cs.washington.edu/courses/cse591b/) Brain-Computer Interfaces Group Meeting

[**CSE 591c: Computational Biology Group Meeting**](http://courses.cs.washington.edu/courses/cse591c/) Computational Biology group meeting. (Was CSE 591CB.)

[**CSE 591d: Database Group Meeting**](http://courses.cs.washington.edu/courses/cse591d/) Database group meeting. (Formerly CSE590DM.)

[**CSE 591e: Reading Seminar in CSE Education & Technology**](http://courses.cs.washington.edu/courses/cse591e/) CSE591E is an informal quarterly seminar focused on Computer Science Education & Technology. Primarily, it consists of reading and discussing papers in this field and related fields. It is open to all students interested in education. (Formerly CSE590ED.)

[**CSE 591f: Robotics Lab Group Meeting**](http://courses.cs.washington.edu/courses/cse591f/) We discuss recent developments in robotics, focusing on probabilistic techniques and multi-robot collaboration.

[**CSE 591g: GRAIL Group Meeting**](http://courses.cs.washington.edu/courses/cse591g/) GRAIL lab meeting and research seminar. The seminar is jointly offered with MSR (Microsoft Research) every other quarter, meeting once a week at MSR for research presentations by UW and MSR personnel. (Formerly CSE590GM.)

[**CSE 591h: Humanoid Robotics Group Meeting**](http://courses.cs.washington.edu/courses/cse591h/) Humaniod Robotics group meeting.

[**CSE 591i: Activity Recognition Group Meeting**](http://courses.cs.washington.edu/courses/cse591i/) Activity Recognition group meeting.

[**CSE 591j: Data Compression Group Meeting**](http://courses.cs.washington.edu/courses/cse591j/) Data Compression group meeting. (Was CSE 590GZ.)

[**CSE 591k: Behavior Recognition & Assisted Cognition Group Meeting**](http://courses.cs.washington.edu/courses/cse591k/) Research in behavior recognition & assisted cognition. (Was CSE 591HK.)

[**CSE 591l: Statistical Analysis and Tools Group Meeting**](http://courses.cs.washington.edu/courses/cse591l/) Statistical Analysis and Tools Group Meeting

[**CSE 591m: HCI Group Meeting**](http://courses.cs.washington.edu/courses/cse591m/) HCI Group Meeting

[**CSE 591n: Configurable Computing Group Meeting**](http://courses.cs.washington.edu/courses/cse591n/) The configurable computing seminar is focused on research on configurable architectures for computing and how to craft programming languages/compilers to leverage these architectures to maximum benefit. Formerly numbered CSE590CC.

[**CSE 591o: SAMPA Group Meeting**](http://courses.cs.washington.edu/courses/cse591o/) SAMPA Group Meeting

[**CSE 591p: Multimedia Group Meeting**](http://courses.cs.washington.edu/courses/cse591p/) Multimedia Group Meeting

[**CSE 591q: Quantum Computing Journal Club**](http://courses.cs.washington.edu/courses/cse591q/) Our group studies many aspects of quantum computing, from ideas about how to build a quantum computer, to the quantum algorithms that will run on these future devices, as well as the greater implications that this intersection of physics and computer science has for both subjects.

[**CSE 591s: Software Engineering Research Group Meeting**](http://courses.cs.washington.edu/courses/cse591s/) Software Engineering Research Group Meeting

**Graduate Courses**

[**CSE 501: Programming Language Analysis And Implementation**](http://courses.cs.washington.edu/courses/cse501/) Design and implementation of compilers and run-time systems for imperative, object-oriented, and functional languages. Intra- and interprocedural analyses and optimizations. Prerequisite: CSE major and CSE 341; recommended CSE 401.

[**CSE 503: Software Engineering**](http://courses.cs.washington.edu/courses/cse503/) Specification, implementation, and testing of large, multiperson, software systems. Topics include abstraction, information hiding, software development environments, and formal specifications. Prerequisite: CSE major and CSE 322, CSE 326, and CSE 378 or equivalents.

[**CSE 504: Advanced Topics In Software Engineering**](http://courses.cs.washington.edu/courses/cse504/) Topics vary but may include software design and evolution, formal methods, requirements specifications, software and system safety, reverse engineering, real-time software, metrics and measurement, programming environments, and verification and validation. Prerequisite: CSE major or permission of instructor.

[**CSE 505: Principles Of Programming Languages**](http://courses.cs.washington.edu/courses/cse505/) Design and formal semantics of modern programming languages, includes functional and object-oriented languages. Prerequisite: CSE major and CSE 341.

[**CSE 506: Advanced Topics In Programming Languages**](http://courses.cs.washington.edu/courses/cse506/) May include functional, object-oriented, parallel, and logic programming languages; semantics for languages of these kinds; type declaration, inference, and checking (including polymorphic types); implementation issues, such as compilation, lazy evaluation, combinators, parallelism, various optimization techniques. Implementation project required. Prerequisite: CSE major, CSE 501 which may be taken concurrently, and CSE 505.

[**CSE 507: Computer-Aided Reasoning for Software**](http://courses.cs.washington.edu/courses/cse507/) Covers theory, implementation, and applications of automated reasoning techniques, such as satisfiability solving, theorem proving, model checking, and abstract interpretation. Topics include concepts from mathematical logic and applications of automated reasoning to the design, construction, and analysis of softwar

[**CSE 510: Advanced Topics In Human-computer Interaction**](http://courses.cs.washington.edu/courses/cse510/) Content varies, including interface issues for networks, embedded systems, education applications, safety and critical systems, graphics and virtual reality, databases, and computer-supported cooperative work.

[**CSE 512: Data Visualization**](http://courses.cs.washington.edu/courses/cse512/) Techniques and algorithms for creating effective visualizations based on principles from graphic design, visual art, perceptual psychology and cognitive science. Topics: data and image models, visual encoding, graphical perception, color, animation, interaction techniques, graph layout, automated design. Lectures, reading and project. Prerequisites: CSE 332 or 373 or equivalent.

[**CSE 513: Disability Inclusion for Technologists**](https://courses.cs.washington.edu/courses/cse513/) Introduction to the future of access technologies. Topics include disability justice; accessibility technology; creating accessible applications and websites; accessibility concerns in a wide variety of computational topics such as security, fabrication, and augmented reality; and the importance of technology in addressing accessibility concerns in the world at large.

[**CSE 515: Statistical Methods In Computer Science**](http://courses.cs.washington.edu/courses/cse515/) Introduction to the probabilistic and statistical techniques used in modern computer systems. Graphical models, probabilistic inference, statistical learning, sequential models, decision theory. Prerequisite: either STAT 341 or STAT 391, and graduate standing in computer science, or permission of instructor.

[**CSE 517: Natural Language Processing**](http://courses.cs.washington.edu/courses/cse517/) Overview of modern approaches for natural language processing. Topics include language models, text classification, tagging, parsing, machine translation, semantics, and discourse analysis.

[**CSE 519: Current Research In Computer Science**](http://www.washington.edu/students/crscat/cse.html#cse519) Weekly presentations on current research activities by members of the department. Only Computer Science graduate students may register, although others are encouraged to attend. Credit/no credit only.

[**CSE 520: Computer Science Colloquium**](https://www.cs.washington.edu/events/colloquia) Weekly public presentations on topics of current interest by visiting computer scientists. Credit/no credit only

[**CSE 521: Design And Analysis Of Algorithms I**](http://courses.cs.washington.edu/courses/cse521/) Principles of design of efficient algorithms: recursion, divide and conquer, balancing, dynamic programming, greedy method, network flow, linear programming. Correctness and analysis of algorithms. NP-completeness. Prerequisite: CSE major and CSE 326 or equivalent. CSE majors only.

[**CSE 522: Design And Analysis Of Algorithms Ii**](http://courses.cs.washington.edu/courses/cse522/) Analysis of algorithms more sophisticated than those treated in 521. Content varies and may include such topics as algebraic algorithms, combinational algorithms, techniques for proving lower bounds on complexity, and algorithms for special computing devices such as networks or formulas. Prerequisite: CSE major and CSE 521.

[**CSE 523: Computational Geometry**](http://courses.cs.washington.edu/courses/cse523/) Algorithms for discrete computational geometry. Geometric computation, range searching, convex hulls, proximity, Vornoi diagrams, intersection. Application areas include VLSI design and computer graphics. Prerequisite: CSE major and CSE 521; recommended: CSE 457 or equivalent.

[**CSE 524: Parallel Algorithms**](http://courses.cs.washington.edu/courses/cse524/) Design and analysis of parallel algorithms: fundamental parallel algorithms for sorting, arithmetic, matrix and graph problems and additional selected topics. Emphasis on general techniques and approaches used for developing fast and efficient parallel algorithms and on limitations to their efficacy. Prerequisite: CSE major and CSE 521.

[**CSE 525: Randomized Algorithms And Probablisitc Analysis**](http://courses.cs.washington.edu/courses/cse525/) Examines algorithmic techniques: random selection, random sampling, backwards analysis, algebraic methods, Monte Carlo methods, and randomized rounding; random graphs; the probabilistic method; Markov chains and random walks; and analysis tools: random variables, moments and deviations, Chernoff bounds, martingales, and balls in bins. Prerequisite: CSE 521 or equivalent; CSE majors only. Offered: WSp.

[**CSE 526: Cryptography**](https://courses.cs.washington.edu/courses/cse526) Introduction to the theoretical foundation of cryptography, teaching the design and application of selected important cryptographic objects, and the mathematical frameworks and methodologies of modern cryptography for formalizing security goals and developing provably secure solutions.

[**CSE 527: Computational Biology**](http://courses.cs.washington.edu/courses/cse527/) Introduces computational methods leveraging artificial intelligence (AI) and machine learning (ML) techniques to understand biological systems and enhance healthcare. Utilizes various AI/ML techniques, including explainable AI, interpretable ML, deep learning, probabilistic graphical models, and causal inference. Explores diverse problem areas such as genetics, epigenomics, transcriptomics, proteomics, imageomics, and electronic health records.

[**CSE 528: Computational Neuroscience**](http://courses.cs.washington.edu/courses/cse528/) Introduction to computational methods for understanding nervous systems and the principles governing their operation. Topics include representation of information by spiking neurons, information processing in neural circuits, and algorithms for adaptation and learning. Prerequisite: elementary calculus, linear algebra, and statistics, or by permission of instructor. Offered: jointly with NEUBEH 528.

[**CSE 529: Computational Genomics**](https://courses.cs.washington.edu/courses/cse529/) Computational and statistical approaches and practices for deriving robust and rigorous insights from modern genomics datasets. Lectures alternate between genomics-inspired problem formulation and foundational statistical and computational approaches for addressing them. In foundational lectures, we will cover basics of statistical inference, hidden confounding factors, causality and causal inference, deep neural networks and interpretation approaches to deep learning models.

[**CSE 529: Neural Control Of Movement: A Computational Perspe**](http://courses.cs.washington.edu/courses/cse529/) Systematic overview of sensorimotor function on multiple levels of analysis, with emphasis on the phenomenology amenable to computational modeling. Topics include musculoskeletal mechanics, neural networks, optimal control and Bayesian inference, learning and adaptation, internal models, and neural coding and decoding. Prerequisite: vector calculus, linear algebra, MATLAB, or permission of instructor. Offered: jointly with AMATH 533; W.

[**CSE 531: Computational Complexity I**](http://courses.cs.washington.edu/courses/cse531/) Deterministic and nondeterministic time and space complexity, complexity classes, and complete problems. Time and space hierarchies. Alternation and the polynomial-time hierarchy. Circuit complexity. Probabilistic computation. Exponential complexity lower bounds. Interactive proofs. Prerequisite: CSE majors only; CSE 322 or equivalent.

[**CSE 532: Computational Complexity Ii**](http://courses.cs.washington.edu/courses/cse532/) Advanced computational complexity including several of the following: circuit complexity lower bounds, #p and counting classes, probabilistically-checkable proofs, de-randomization, logical characteristics of complexity, communication complexity, time-space tradeoffs, complexity of data structures. Prerequisite: CSE majors only; Recommended: CSE 531.

[**CSE 533: Advanced Topics In Complexity Theory**](http://courses.cs.washington.edu/courses/cse533/) An in-depth study of advanced topics in computational complexity. Prerequisite: CSE major.

[**CSE 534: Quantum Information and Computation**](http://courses.cs.washington.edu/courses/cse534) Introduction to quantum information and computation. Qubits, quantum gates, and measurements. Entanglement and non-locality. Density matrix formalism. Quantum algorithms: Simon's algorithm, Grover search, Shor's factoring, and Hamiltonian Simulation. Quantum error-correction.

[**CSE 535: Theory of Optimization and Continuous Algorithms**](http://courses.cs.washington.edu/courses/cse535/)Theoretical foundations of convex optimization and continuous algorithms. First-order methods, rates of convergence, and acceleration; gradient, subgradient, and mirror descent. Randomization, stochastic descent, leverage scores and sampling. Interior point methods. Linear systems in convex optimization. Algorithmic applications.

[**CSE 536: Theory Of Distributed Computing**](http://courses.cs.washington.edu/courses/cse536/) Formal approaches to distributed computing problems. Topics vary, but typically include models of distributed computing, agreement problems, impossibility results, mutual exclusion protocols, concurrent reading while writing protocols, knowledge analysis of protocols, and distributed algorithms. Prerequisite: CSE major.

[**CSE 540: Discrete System Simulation**](http://courses.cs.washington.edu/courses/cse540/) Principles of simulation of discrete, event-oriented systems. Model construction, simulation and validation. Distributed and parallel simulation techniques. Basic statistical analysis of simulation inputs and outputs. Use of C, an object-oriented language, and S, a statistical analysis package. Prior familiarity with the concepts of probability and statistics desirable.

[**CSE 541: Interactive Learning**](https://courses.cs.washington.edu/courses/cse541) Foundations and methods of interactive machine learning including multi-armed bandits, active learning, and adaptive experimental design. Stochastic and adversarial K-armed bandits, structured bandits, and contextual bandits. Online learning; adaptive data collection for supervised learning.

[**CSE 543: Deep Learning**](http://courses.cs.washington.edu/courses/cse543/) Foundations of modern deep learning and its applications. Topics include various forms of deep neural networks architectures such as convolution neural networks, recurrent networks, attention mechanism, training and optimization algorithms and their applications in domains such as computer vision, robotics, and natural language processing. Recommended: comfortable with programming; and working knowledge of linear algebra (MATH 208); vector calculus (MATH 126); probability and statistics (CSE 312 and STAT 390); algorithms (CSE 421); and machine learning (CSE 446 or CSE 546)

[**CSE 544: Principles Of Database Systems**](http://courses.cs.washington.edu/courses/cse544/) The relational data model: SQL, Relational Algebra, Relational Calculus, discussion of other data models. Database systems: indexes, query execution and optimization, database statistics, parallel databases, MapReduce. Database theory: datalog and extensions with negation, query complexity, query containment and equivalence, bounded tree width. Miscellaneous: transactions, data provenance, data privacy, probabilistic databases.

[**CSE 546: Machine Learning**](http://courses.cs.washington.edu/courses/cse546/) Explores methods for designing systems that learn from data and improve with experience. Supervised learning and predictive modeling; decision trees, rule induction, nearest neighbors, Bayesian methods, neural networks, support vector machines, and model ensembles. Unsupervised learning and clustering. Prerequisite: either STAT 341, STAT 391, or equivalent, or permission of instructor.

[**CSE 547: Machine Learning for Big Data**](http://courses.cs.washington.edu/courses/cse547/) Machine Learning and statistical techniques for analyzing datasets of massive size and dimensionality. Representations include regularized linear models, graphical models, matrix factorization, sparsity, clustering, and latent factor models. Algorithms include sketching, random projections, hashing, fast nearest-neighbors, large-scale online learning, and parallel (Map-reduce, GraphLab). Prerequisite: either STAT 535 or CSE 546. This course is cross-listed as STAT 548.

[**CSE 548: Computer Systems Architecture**](http://courses.cs.washington.edu/courses/cse548/) Notations for computer systems. Processor design (single chip, look-ahead, pipelined, data flow). Memory hierarchy organization and management (virtual memory and caches). Microprogramming. I/O processing. Multiprocessors (SIMD and MIMD). Prerequisite: CSE major and CSE 451.

[**CSE 549: High-performance Computer Architectures**](http://courses.cs.washington.edu/courses/cse549/) Algorithm design, software techniques, computer organizations for high-performance computing systems. Selected topics from: VLSI complexity for parallel algorithms, compiling techniques for parallel and vector machines, large MIMD machines, interconnection networks, reconfigurable systems, memory hierarchies in multiprocessors, algorithmically specialized processors, data flow architectures. Prerequisite: CSE major and CSE 548 or permission of instructor.

[**CSE 550: Computer Systems**](http://courses.cs.washington.edu/courses/cse550/) Explores computer system design, implementation, and evaluation. Covers principles, techniques, and examples related to the construction of computer systems, including concepts that span network systems, operating systems, web servers, parallel computing, and databases. Prerequisite: CSE 451.

[**CSE 551: Operating Systems**](http://courses.cs.washington.edu/courses/cse551/) Operating system design and construction techniques. Concurrent programming, operating system kernels, correctness, deadlock, protection, transaction processing, design methodologies, comparative structure of different kinds of operating systems, and other topics. Prerequisite: CSE major and CSE 451.

[**CSE 552: Distributed And Parallel Systems**](http://courses.cs.washington.edu/courses/cse552/) Principles, techniques, and examples related to the design, implementation, and analysis of distributed and parallel computer systems. Prerequisite: CSE major and CSE 551.

[**CSE 553: Real-time Systems**](http://courses.cs.washington.edu/courses/cse553/) Design and construction of software for real-time computer systems. Software architectures. Requirements and specification methods. Scheduling algorithms and timing analysis. Real-time operating systems. Real-time programming languages. Selected case studies. Prerequisite: CSE major and CSE 451.

[**CSE 556: Computational Fabrication**](https://courses.cs.washington.edu/courses/cse556/) Overview of the computational tools and concepts used throughout the modern pipeline for computational fabrication, including topics such as hardware abstraction languages, geometry processing fundamentals, physics-based simulation, optimization techniques, data-driven design methods, and algorithms for high-performance interactive applications.

[**CSE 557: Computer Graphics**](http://courses.cs.washington.edu/courses/cse557/) Introduction to image synthesis and computer modeling, emphasizing the underlying theory required for undertaking computer graphics research. Topics include color theory, image processing, affine and projective geometry, hidden-surface determination, photorealistic image synthesis, advanced curve and surface design, dynamics, realistic character animation. Prerequisite: CSE major, solid knowledge of linear algebra.

[**CSE 558: Special Topics In Computer Graphics**](http://courses.cs.washington.edu/courses/cse558/) Advanced topics in computer graphics not treated in CSE 557. Topics vary from year to year but typically include advanced aspects of image synthesis, animation, and 3D photography. Prerequisite: CSE major and CSE 557 or permission of instructor.

[**CSE 561: Computer Communications And Networks**](http://courses.cs.washington.edu/courses/cse561/) Fundamentals of data transmission: coding, message formats, and protocols. Organization of computer networks. Examples of network implementations. Prerequisite: CSE or E E major and CSE 451 or equivalent.

[**CSE 564: Computer Security And Privacy**](http://courses.cs.washington.edu/courses/cse564/) Examines the fundamental of computer security including: human factors; attack detection, measurements, and models; cryptography and communications security; system design and implementation; and side channels.

[**CSE 567: Principles Of Digital Systems Design**](http://courses.cs.washington.edu/courses/cse567/) Principles of logic design, combinational and sequential circuits, minimization techniques, structured design methods, CMOS technology, complementary and ratioed gates, delay estimation and performance analysis, arithmetic circuits, memories, clocking methodologies, synthesis and simulation tools, VLSI processor architecture. Prerequisite: CSE major and basic knowledge of logic design.

[**CSE 568: Introduction To VLSI Systems**](http://courses.cs.washington.edu/courses/cse568/) Introduction to CMOS technology and circuit design; combinational logic-design alternatives; register-design and system-clocking methodologies; datapath and subsystem design; VLSI system-design methodologies; CAD tools for synthesis, layout, simulation, and validation; design of a complex VLSI chip. Prerequisite: CSE 567 or permission of instructor. CSE majors only.

[**CSE 571: Probabilistic Robotics**](http://courses.cs.washington.edu/courses/cse571/) This course introduces various techniques for Bayesian state estimation and its application to problems such as robot localization, mapping, and manipulation. The course will also provide a problem-oriented introduction to relevant machine learning and computer vision techniques.

[**CSE 573: Artificial Intelligence**](http://courses.cs.washington.edu/courses/cse573/) Broad introduction to the science of automated rational decision-making by machines. Key approaches include search, Markov decision processes, graphical models, reinforcement learning, and supervised learning. Considers a wide variety of application domains (e.g., natural language processing, computer vision, robotics, games) for decision making.

[**CSE 574: Explainable Artificial Intelligence**](http://courses.cs.washington.edu/courses/cse574/) Approaches to enhancing the interpretability and transparency of complex machine learning models, encompassing both inherently interpretable models and post hoc explanation methods. Explores a spectrum of techniques, ranging from feature attributions and their evaluation metrics to counterfactual explanations, concept-based explanations, instance explanations, and collaboration between humans and artificial intelligence.

[**CSE 576: Computer Vision**](http://courses.cs.washington.edu/courses/cse576/) Overview of computer vision, emphasizing the middle ground between image processing and artificial intelligence. Image formation, preattentive image processing, boundary and region representations, and case studies of vision architectures. Prerequisite: Solid knowledge of linear algebra, good programming skills, CSE or E E major or permission of instructor. Offered: jointly with E E 576.

[**CSE 577: Special Topics In Computer Vision**](http://courses.cs.washington.edu/courses/cse577/) Topics vary and may include vision for graphics, probabilistic vision and learning, medical imaging, content-based image and video retrieval, robot vision, or 3D object recognition. Prerequisite: CSE/E E 576 or permission of instructor. Offered: jointly with E E 577.

[**CSE 578: Convex Optimization**](http://courses.cs.washington.edu/courses/cse578) Basics of convex analysis: Convex sets, functions, and optimization problems. Optimization theory: Least-squares, linear, quadratic, geometric and semidefinite programming. Convex modeling. Duality theory. Optimality and KKT conditions. Applications in signal processing, statistics, machine learning, control communications, and design of engineering systems. Prerequisite: A A 510, CHEM E 510, E E 510, or M E 510. Offered: jointly with A A 578/E E 578/M E 578;

[**CSE 579: Intelligent Control Through Learning & Optimization**](http://homes.cs.washington.edu/~todorov/courses/amath579/) Design or near-optimal controllers for complex dynamical systems, using analytical techniques, machine learning, and optimization. Topics from deterministic and stochastic optimal control, reinforcement learning and dynamic programming, numerical optimization in the context of control, and robotics. Prerequisite: vector calculus; linear algebra, and Matlab. Recommended: differential equations; stochastic processes, and optimization. Offered: jointly with AMATH 579.

[**CSE 580: Computing for Social Good**](https://courses.cs.washington.edu/courses/cse580/)Explorations of computing for social good projects. Topics include best practices, exemplars of success and failure, differing theories of change and perspectives on technology's role in progress. Students are expected to develop a solution grounded in their own perspective of social good.

[**CSE 581: Computer Ethics**](http://courses.cs.washington.edu/courses/cse581/) Ethical and philosophical approaches to thinking critically about computing technology. Tools and strategies for considering unintended consequences. Responsibilities of computer science researchers and practitioners. Application to various technology areas.

[**CSE 582: Ethics in Artificial Intelligence**](https://courses.cs.washington.edu/courses/cse582/) Real-world people-facing artificial intelligence (AI) applications, ethical implications in their design, and technical solutions to mitigate ethical risks. Topics in the intersection of AI, ethics, and computing for social good, centered around foundational and recent research. Foundations of research ethics, case studies in ethical AI, ethical challenges in developing intelligent systems, and machine learning approaches to address these issues. Prerequisite: CSE 446 or CSE 546.

[**CSE 583: Software Development for Data Scientists**](http://courses.cs.washington.edu/courses/cse583/) Provides students outside of CSE with a practical knowledge of software development that is sufficient to do graduate work in their discipline. Modules include Python basics, software version control, software design, and using Python for machine learning and visualization.

[**CSE 586: Introduction To Synthetic Biology**](http://courses.cs.washington.edu/courses/cse586/) Studies mathematical modeling of transcription, translation, regulation, and metabolism in cell; computer aided design methods for synthetic biology; implementation of information processing, Boolean logic and feedback control laws with genetic regulatory networks; modularity, impedance matching and isolation in biochemical circuits; and parameter estimation methods. Prerequisite: either MATH 136 or MATH 307, AMATH 351, or CSE 321 and MATH 208 or AMATH 352. Offered: jointly with BIOEN 523/E E 523.

[**CSE 587: Advanced Systems And Synthetic Biology**](http://courses.cs.washington.edu/courses/cse587/) Introduces advanced topics in systems and synthetic biology. Topics include advanced mathematical modeling; computational standards; computer algorithms for computational analysis; and metabolic flux analysis, and protein signaling pathways and engineering. Prerequisite: either BIOEN 523,E E 523, or CSE 586. Offered: jointly with BIOEN 524/E E 524; W.

[**CSE 589: Software Entrepreneurship**](https://courses.cs.washington.edu/courses/cse589/) A case- and project-based course, focusing on starting a software or hardware company. Guest entrepreneurs, lawyers, and financiers discuss market identification and analysis, planning the business, financing, and typical operating and administrative problems. Offered: jointly with CSE P 589/ENTRE 532.

[**CSE 599a1: Entrepreneurship: Company-Building from Formation to Successful Exit**](https://courses.cs.washington.edu/courses/cse599a1) This course is open to Allen School undergraduates, combined BS/MS students, Professional Masters Program students, and Ph.D. students, as well as to Foster School MBA and EMBA students, students in Interaction Design, graduate students in Human Centered Design & Engineering, and students in the Master of Human-Computer Interaction and Design program - all by permission of the instructors in order to ensure balance among the participants. There will be no auditing - everyone needs to be all-in. And project teams will form early - if you hang on for a week or two and then bail, you'll be letting others down, so please don't do this. Application link: https://catalyst.uw.edu/webq/survey/lazowska/414693

[**CSE 599A1: Entrepreneurship**](http://courses.cs.washington.edu/courses/cse599a1/) Company-Building from Formation to Successful Exit. This course is about entrepreneurship and specifically about starting, growing, managing, leading, and ultimately exiting a new venture. Offered jointly with CSE 490A, CSE P 590, ENTRE 532.

[**CSE 599F1: Constraint Programming**](https://courses.cs.washington.edu/courses/cse599f1/) Design, implementation, and use of constraint programming languages.

[**CSE 599H: Computer Science for Social Good**](http://make4all.org/curricula/computer-science-for-social-good/) The goal of this course is to use computer science tools to tackle real world projects in a productive fashion. The Fall 2017 edition of this course focused on 3D printing and its applications for accessibility, education, sustainability, and the developing world.

[**CSE 599J1: Selected Topics in Computational Fabrication**](https://courses.cs.washington.edu/courses/cse599j1/) This course explores research problems in computational design for manufacturing. Topics include additive manufacturing, subtractive manufacturing, knitting, architecture, mechanisms, robotics, aerodynamics, machine learning, and programming languages.

[**CSE 600: Independent Study Or Research**](http://www.cs.washington.edu/education/courses/) Credit/no credit only.

[**CSE 700: Masters Thesis**](http://www.cs.washington.edu/education/courses/) Credit/no credit only.

[**CSE 800: Doctoral Dissertation**](http://www.cs.washington.edu/education/courses/) Credit/no credit only.

**Graduate Seminars**

[**CSE 590: Research in CS**](http://courses.cs.washington.edu/courses/cse590k/) Seminar for newer CSE graduate students introducing research opportunities in the CSE department.

[**CSE 590a: Research Seminar**](http://courses.cs.washington.edu/courses/cse590a/) Current research in artificial intelligence at UW CSE.

[**CSE 590b: Computer Graphics Seminar**](http://courses.cs.washington.edu/courses/cse590b/) Computer Graphics Seminar

[**CSE 590c: Reading and Research in Computational Biology**](http://courses.cs.washington.edu/courses/cse590c/) Goals: A taste of current research in Computational Biology (local and non-) + critical reading of literature + presentation skills. Students, with faculty advice, pick and present CompBio papers from recent journals/conferences. Students & faculty also present their own research (mostly in Spring, but may be sprinkled throughout, depending on schedules). Background knowledge of biology is not assumed; come learn!

[**CSE 590c1: Change Seminar**](http://courses.cs.washington.edu/courses/cse590c1/) Change Seminar

[**CSE 590d: Live Programming: Concepts and Applications, including Musical Performance**](https://courses.cs.washington.edu/courses/cse590d/20au/) The seminar addresses an “extreme” form of programming in which programs are edited while they are running. We consider a mix of issues: semantic, language-design, implementation, as well as applications, especially in musical improvisation and performance. Participants read/present/discuss papers, and try out systems. Most registered students will summarize and lead discussion on one or two papers or systems during the quarter.

[**CSE 590e: Computer Science Education Seminar**](http://courses.cs.washington.edu/courses/cse590e/) Understanding themes and techniques of research in computer science education. Discuss the application of those concepts to practice in teaching. Focuses on giving new participants breadth of coverage while complementing what returning students have seen. Seminar is designed to be accessible to new participants with limited teaching or education research experience. Featured 20au (and beyond): Exploring anti-racist CS education research and practice.

[**CSE 590em: Environmental Modelling Seminar**](http://courses.cs.washington.edu/courses/cse590em/) Environmental Modelling Seminar

[**CSE 590es: Special Topics in Embedded Systems**](http://courses.cs.washington.edu/courses/cse590es/) Seminar or project course where students investigate different aspects of embedded systems. Topics change with each offering. The objective is to expose students to the different nature of task-specific computing devices and their communication methods as opposed to general-purpose computing systems and networks.

[**CSE 590f: Computing and the Developing World Seminar**](http://courses.cs.washington.edu/courses/cse590f/) A focused reading seminar looking at the latest papers and ideas across the broader ICTD (Information & Communication Tech for Development) and C4SG (Computing for Social Good) space. We use the first meeting to pick readings as a group, which can range from current conference papers, influential historical papers, or books/book chapters. (Formerly numbered CSE590DW.)

[**CSE 590g: Architecture Lunch**](http://courses.cs.washington.edu/courses/cse590g/) A reading seminar in which we discuss current research published in the key architecture conferences or advanced subtopics in computer architecture.

[**CSE 590h: Human-Computer Interaction Seminar**](http://courses.cs.washington.edu/courses/cse590h/) Open to all people who are interested in/studying HCI!

[**CSE 590j: DUB Seminar**](http://courses.cs.washington.edu/courses/cse590j/) An interdisciplinary (Data science, HCI, Science Technology and Society Studies, and more), cross-campus human-computer interaction seminar. https://dub.uw.edu

[**CSE 590l: Computer networks seminar**](http://courses.cs.washington.edu/courses/cse590l/) Computer networks seminar.

[**CSE 590m2: Machine Learning Reading Group**](http://courses.cs.washington.edu/courses/cse590m2/) Readings in the research frontier in Machine Learning. Previously labelled CSE590PD.

[**CSE 590n: Software Engineering Seminar**](http://courses.cs.washington.edu/courses/cse590n/) A seminar for those interested in general topics in software engineering. It's run by David Notkin and is historically quite informal.

[**CSE 590nc: Neural Computation Seminar**](http://courses.cs.washington.edu/courses/cse590nc/)

Neural computation and learning. Discussions of classic as well as recent papers in neural computing and neuroscience. Participants select one or more papers from a reading list and will lead the corresponding discussion meetings. Specific topics covered include:

* Supervised and unsupervised learning
* Reinforcement learning and imitation learning
* Bayesian inference and relationship to neural networks
* Recurrent and hierarchical networks
* Applications in computer vision, robotics, and brain-computer interfaces

[**CSE 590o: Parallel Programming Environments**](http://courses.cs.washington.edu/courses/cse590o/) Parallel Programming Environments

[**CSE 590p: Programming Systems**](http://courses.cs.washington.edu/courses/cse590p/) Programming Systems

[**CSE 590q: Database Seminar**](http://courses.cs.washington.edu/courses/cse590q/) We choose a different topic each quarter. Recent examples of topics: “Responsible Data Management”, “Learned Database Systems”, “Deep Learning Meets DB”, “Transactions!”. We present 1-2 papers per seminar. Presentations are either by one or two students. If you sign up for more than 1 credit then you must present a paper. (Was CSE 590DB.)

[**CSE 590qc: Quantum Computing**](http://courses.cs.washington.edu/courses/cse590qc/) 590QC covers the material from Mark Oskin's quantum computing course from Spring 2002, meeting once a week to discuss the lecture notes and readings, and homework is encouraged to whatever extend the student finds helpful. See [this syllabus](http://courses.cs.washington.edu/courses/cse590mo/02sp/).

[**CSE 590r: Robotics Colloquium**](http://www.cs.washington.edu/news_events/robotics_colloquia/) The Robotics Colloquium features talks by invited and local researchers on all aspects of robotics, including control, perception, machine learning, mechanical design, and interaction.

[**CSE 590s: Systems Seminar**](http://courses.cs.washington.edu/courses/cse590s/) Discussion of recent research in operating systems, distributed systems, and networks.

[**CSE 590t: Society and Technology**](http://courses.cs.washington.edu/courses/cse590t/) From large-scale systems for searching the web to the datasets that machine learning practitioners utilize to train their models, our collective cultural heritage is in many ways the substrate of computer science. Indeed, cultural heritage practitioners including humanists, librarians, and archivists have been influential in shaping the discourse surrounding the sociotechnical implications of computing. This course explores various topics within computer science through the lens of cultural heritage: data visualization, human-AI interaction, search & discovery, crowdsourcing, web archiving, design & UX, and classification. The goals of this course are two-fold: first, to survey these topics in computer science, and second, to explore how they manifest within the context of cultural heritage. We will cover one topic every week, with the first meeting devoted to the CS-oriented literature for the topic, and the second meeting devoted to the sociotechnical implications of the topic in practice. During these second meetings, we will speak with cultural heritage practitioners at institutions across the country to learn about the roles of computing in their work, research, and stewardship. This seminar is accessible to those without any formal training in computer science. Ben welcomes students across a range of disciplines, including library and information science, the humanities, and computer science & engineering. Apply for enrollment here: https://forms.gle/RoYjGxSZPwtW6R336

[**CSE 590u: Topics in Ubiquitous Computing**](http://courses.cs.washington.edu/courses/cse590u/) A overview of current research in ubiquitous computing (Ubicomp) designed to bring together researchers from different areas of Computer Science. (Was CSE 590UC.)

[**CSE 590v: Computer Vision**](http://courses.cs.washington.edu/courses/cse590v/) Topics include recent advances in computer graphics and vision. One or two speakers will present each time, talking about their researches or recent papers. There will also be several guest lectures introducing popular topics in this area.

[**CSE 590w: Accessibility Research Seminar**](http://courses.cs.washington.edu/courses/cse590w/) The seminar is for students and faculty members to explore research in accessible computing for people with disabilities in the context of human-computer interaction (HCI). The seminar consists of short student presentations of current research results, followed by discussion and critical evaluations the research. https://create.uw.edu/quarterly-seminar-on-accessibility-research

[**CSE 590x: How to Navigate the PhD program**](https://courses.cs.washington.edu/courses/cse590x/) How to Navigate the PhD program

[**CSE 590y: Security Seminar**](http://courses.cs.washington.edu/courses/cse590y/) Security Seminar

[**CSE 590z: Theory Seminar**](http://courses.cs.washington.edu/courses/cse590z/) Talks from a mix of local and external speakers on the latest and greatest in CS theory; from algorithms to complexity theory, from optimization to cryptography, from information theory to game theory. All your theory needs under one roof. https://theory.cs.washington.edu

[**CSE 599c: Topics in Data Management Systems**](http://courses.cs.washington.edu/courses/cse599c/) Topics in Data Management Systems

[**CSE 599g: Selected Topics in Computer Security**](http://courses.cs.washington.edu/courses/cse599g/) Selected Topics in Computer Security

[**CSE 599j: Human System and Control for Neurobotics**](http://courses.cs.washington.edu/courses/cse599j/) Human System and Control for Neurobotics

[**CSE 599n: Deep Learning for Neuroscience**](http://courses.cs.washington.edu/courses/cse599n/23wi) Brains are remarkably complex, massive networks of interconnected neurons that underlie our abilities to intelligently sense, reason, learn, and interact with our world. Technologies for monitoring neural activity in the brain are revealing rich structure within the coordinated activity of these interconnected populations of neurons. In this course, we will discuss deep learning models that can be applied toward 1) understanding how neural activity in the brain gives rise to intelligent behavior and 2) designing algorithms for brain-interfacing biomedical devices. Topics will focus around variational autoencoders and recurrent neural networks, along with their probabilistic foundations from classical machine learning. Coursework will include readings from the deep learning and computational neuroscience literature, programming assignments, and a final modeling project applied to neural population data.

[**CSE 599o: Context-Aware Mobile Computing**](http://courses.cs.washington.edu/courses/cse599o/) Context-Aware Mobile Computing

[**CSE 599q: Topics in Multicore Programmability**](http://courses.cs.washington.edu/courses/cse599q/) Topics in Multicore Programmability

[**CSE 599t: Probabilistic and Statistical Databases**](https://courses.cs.washington.edu/courses/cse599t/) Probabilistic and Statistical Databases

[**CSE 599u: Advanced Topics in Ubiquitous Computing**](http://courses.cs.washington.edu/courses/cse599u/) Advanced Topics in Ubiquitous Computing

[**CSE 599w: Operating Systems and the Web**](http://courses.cs.washington.edu/courses/cse599w/) Operating Systems and the Web