CS Undergraduate Course Descriptions

EN 1-CCS The Craft of Computer Science

While it might be obvious why we need research in biology or history, Computer Science research is necessary to reinvent the field and to drive discoveries across many disciplines. This course will teach you the foundations of research. Students will work with a faculty mentor and a student group on a research project. Research topics include machine learning, computer security, quantum computing, human-robotics interaction, computational biology, computational geometry, and others. The course will cover topics including identifying and formulating research problems, reading and evaluating research papers, literature searching, self-guided learning, designing research studies, and data analysis. Students will practice working in a team, goal setting, activity logging, and communicating with others. This is a non-coding class. No prior coding or CS experience is required. Student groups are expected to develop a research proposal by the end of the semester and to be well-prepared to participate in future Computer Science research experiences.

EN 1-ECS Exploring Computer Science

How does one translate a strategy written in English into executable computer code written in the C++ programming language or some other high-level language? How does one evaluate two different strategies for solving the same problem? What sorts of problems are solved well by computers? How can the solutions be displayed graphically? This course is intended for those who have NO programming experience. This sampling of various topics within the field of computer science will give the student a taste of the broader spectrum that constitutes computer science. The course will include a general introduction to the field of computer science, to the emacs editor, to the C++ programming language, and to the Linux operating system.

Prerequisite: A sincere interest in learning more about computer science and NO prior programming experience.

EN 1-EK Enginnering in the Kitchen

In this course, we will explore engineering through the lens of food and kitchen gadgets. During the semester, we will disassemble every electrified food-preparation device we can get our hands on, learn how they work, and use our newfound skills to build a few of our own. Along the way, you'll analyze and design basic electrical circuits, program microcontrollers to take measurements and respond to them, log data to answer questions about cooking, and connect the Things you build to the Internet. We'll also explore some of the complex social and ethical issues at the intersection of technology and food: does a cloud-connected refrigerator make us more efficient, or more lazy, or does it just result in more e-waste? And what responsibility do engineers have when working with something so deeply human as food?

Prerequisite: First Year Students Only

EN 1-ICD Intro to Computational Design

Can we be inspired by caterpillars to design useful robots? Can we use computers to deduce the neuro-mechanical commands that make a caterpillar move forward? What do caterpillars and soft-tissue robots have in common? This class brings such interdisciplinary research experiences to first-year Engineering students through the lens of computational design. We learn in this course about computational modeling, the meaning of computational design and how it can help engineers in making best design decisions. The course will use MATLAB as a computational platform, and cover fundamentals such as a solution space, design decision variables, constraints, optimal points within the space and searching the design space using efficient algorithms, exhaustive enumeration, and approximate algorithms.

EN 1-SR Simple Robotics

Introduction to robot construction, programming, computer vision, event-based programming, artificial intelligence, and elementary controls. Basic principles of robotics for students with minimal or no prior programming/building background. In-class competition-based laboratories and hands-on group projects using the LEGO MINDSTORMS platform.

COMP 5-BTN Beyond Translation

Digital Analysis and Publication of Historical Sources for a Global Audience. All meetings are available via Zoom; some classes include an in person component (primarily in the opening six weeks of the semester).

COMP 5-ECS Exploring Computer Science

How does one translate a strategy written in English into executable computer code written in the C++ programming language or some other high-level language? How does one evaluate two different strategies for solving the same problem? What sorts of problems are solved well by computers? How can the solutions be displayed graphically? This course is intended for those who have NO programming experience. This sampling of various topics within the field of computer science will give the student a taste of the broader spectrum that constitutes computer science. The course will include a general introduction to the field of computer science, to the emacs editor, to the C++ programming language, and to the Linux operating system.

Prerequisite: A sincere interest in learning more about computer science and NO

prior programming experience.

COMP 5-TCS Teaching Computer Science

This course will prepare undergraduates to function effectively and efficiently as undergraduate teaching assistants. Through this course, students will learn pedagogical techniques that match learner needs; discuss ethical and social concerns that UTAs face in the course of a semester; and problem solve together issues that arise as teaching assistants. This course is designed in a learner centered model requiring your active and engaged participation. Through your willingness to share your experiences and expertise and your collaboration with your fellow UTA we will together construct meaningful solutions to difficulty situations. Faculty from Computer Science will participate in some of the sessions as co-facilitators. Students will be expected to complete short readings; keep a reflective blog of your learning as a teacher; give a short final presentation on a topic of interest that you want to explore in more depth to help you in your TA class.

COMP 55-CYB Cybersecurity and Cyberwar

Crosslisted as PS 188.

Interdisciplinary analysis of cybersecurity in the United States and other countries, intended to introduce engineering students to policymaking and intelligence aspects of cybersecurity and liberal arts students to the technical constraints of computer networks and software. Hands-on activities including packet analysis, exploiting a vulnerable system, password cracking, social engineering, reconnaissance, and malware analysis. Examination of state and non-state actors engaged in cyber-espionage, counterintelligence, deterrence, and offensive cyber operations. Guest speakers from private sector, civil liberties groups, and intelligence community.

Prerequisite: PS 61: Introduction to International Relations (for PS and IR majors) or COMP 15: Data Structures (for CS majors in A&S or SOE)

COMP 125 Numerical Analysis

(Cross-listed as MATH 125.) Analysis of algorithms involving computation with real numbers. Interpolation, methods for solving linear and nonlinear systems of equations, numerical integration, numerical methods for solving ordinary differential equations.

Prerequisite: Recommendations: MATH 51 and programming ability in a language such as C, C++, Fortran, or Matlab.

COMP 150-AA Assistive Algorithms

This course will focus on computational approaches to providing assistance to human users: sensing their behaviors, modeling their goals and plans, and intelligently choosing appropriate (including socially appropriate) actions. We will study these challenges through a series of case studies on state-of-the-art approaches to these problems, from smart prosthetics, to smart home assistants, to socially assistive robotics. This course requires some prior programming experience and an exposure to basic algorithms (ME 80 OR (COMP 15 and COMP/MATH 61) OR grad standing), and will focus on the practice of designing computational assistive systems, with a small project and skill-building through homework assignments and exams.

Prerequisite: ME 80 OR (COMP 15 and COMP/MATH 61) OR grad standing

COMP 150-AAC Accessible and Assistive Computing

This course will focus on developing accessible and assistive computational systems. That is, systems that are usable by and helpful to users with disabilities. In addition to learning about accessible and universal design (for example for web accessibility) we will also discuss computational approaches to providing assistance to human users: sensing their behaviors, modeling their goals and plans, and intelligently choosing appropriate (including socially appropriate) actions. We will study these challenges through a series of case studies on state-of-the-art approaches to these problems, from web accessibility and universal design to socially assistive robotics.

This course requires some prior programming experience and an exposure to basic algorithms (ME 80 OR (COMP 15 and COMP/MATH 61) OR grad standing), and will focus on the practice of designing accessible and/or assistive systems, with a small project and skill-building through homework assignments and exams.

COMP 150-ADS Algorithms and Data Structures 2

This course offers an opportunity to expand your knowledge on various topics involving algorithms, data structures and graphs. Often these topics are intertwined; e.g., to create efficient algorithms, it may be useful to design data structures or use existing ones. We will cover a range of topics, such as path approximation, spanners, network flow, linear programming, all-pairs shortest paths, near-planarity, Fibonacci heaps and Quake heaps, balanced trees (Splay, WAVL, Scapegoat, BB-alpha), skip lists, fractional cascading, high-dimensional range counting, graph coloring, the probabilistic method, string matching, etc. These are topics that are useful to know, as one prepares for advanced interviews and/or further graduate work. As an elective, this course will aim to let each student focus more on topics that they are interested in. Evaluation will be primarily based on participation and a project.

Prerequisite: Completion of COMP 160 or permission of instructor.

COMP 150-PRH Probabilistic Robotics for HRI

This graduate-level course will introduce various techniques for probabilistic state estimation and examine their application to problems such as robot localization, mapping, perception, and planning in the context of Human-Robot Interaction. The course will also provide a problem-oriented introduction to relevant machine learning and computer vision techniques that are commonly used by robots interacting with humans. Topics include: Overview of mobile robotics (hardware, software architectures, sensors), probabilistic models of sensing and acting, Bayesian state estimation and filtering (e.g., Kalman and particle filters), localization and mapping, computer vision for robot perception (e.g., human activity recognition), and models of robot decision making and learning (e.g., Markov decision processes, reinforcement learning). The main components of the course include several programming assignments along with a final project. You will be able to use Turtlebot2 robots for homework and projects, along with any other robots (or robot simulators) you have access to through your current research activities.

Prerequisite: The course is programming-intensive and at a minimum, you should have 2+ years of solid C++ programming experience. Formal background in probability, statistics, and linear algebra is strongly recommended. For CS undergraduates, you are expected to have taken COMP 40 and another robotics course (e.g,, COMP 50: Autonomous Intelligent Robotics). For ME undergraduates, you are expected to have taken ME 80 Controls. If you're unsure whether you meet the requirements, talk to your instructor to decide if this course is right for you.

COMP 150-SWT Software Testing

In this course we will review the traditional software testing techniques that are applicable to any software product, as well as learn techniques for the paradigm of test-driven development. Continuous delivery/DevOps and its impact on testing will be discussed. We will also discover how innovative companies are able to build testing and quality into every stage of the development process and deliver a multitude of releases with a relatively small testing organization. We will practice test creation and testing techniques through assignments and projects. There will also be an option to practice behavioral development and testing framework Cucumber.

COMP 150-UQC Uncertainty Quantification CPU

TBA

COMP 150-VM Simple Virtual Machines

COMP 152-EAI Explainable Artificial Intelligence

As machine learning systems are increasingly being considered for employments in contexts where their decisions can have real (positive and negative) impacts on people's lives, scholars and activists have increasingly raised concerns about the transparency and accountability of such systems and their designers. The field of explainable artificial

intelligence (XAI) has sprung up in recent years to address some of these concerns by developing systems which can explain the reasons behind their decisions and outputs to users. In this course, we explore this up-and-coming subfield of AI by learning the basic concepts and ideas, and then discussing recent research papers in the field. We will study both the central ideas and concrete systems constructed using these ideas. We will also think more broadly about the assumptions underlying the field of XAI and the ethical debates within the field.

COMP 152-MLLA Machine Learning with Limited Annotation

This course will cover the latest research on machine learning in settings where labeled data is not available. In such settings, classical methods are not necessarily the best approaches for algorithms to learn from data. Students will lead and participate in discussions on recent research publications. Students will also complete a research project applying these new methods. In addition to the specific knowledge about the topic area, students will also gain experience reading and understanding research literature, planning and executing research projects, and communicating about machine learning research. Students are expected to have taken COMP 135 or have equivalent experience, broadly defined.

We will read about ongoing research on how to solve the challenging task of doing machine learning with limited labeled data. We will read a lot of papers and try to understand the problems they solve and their methods. You will do a mini research project during the semester where you either apply an existing method to an application you are interested in or work on developing and evaluating new methods.

COMP 152-NLP Natural Language Processing

An enormous amount of text (news articles, weblog, tweets) is created every day. Natural language processing transforms text into presumably useful data structures, enabling many applications such as real-time event tracking and question answering. In this course, we will study the mathematics and algorithms in NLP to better understand how they do what they do. We will cover a wide range of text analysis methods, include word level (topic and sentiment analysis), syntactical (grammars and parsing), semantic (meanings of words and phrases), and discourse (pronoun resolution and text structure). We will cover both rule-base systems and statistical models. We will code several algorithms applying what we learn in hands-on projects. We will come away with a deeper understanding of how text is processed by a computer.

Prerequisite: Completion of COMP 15, COMP 61, linear algebra (MATH 70, MATH 72, or equivalent), and statistics (ES 56, EE 24, or equivalent); or consent of instructor. Completion of COMP 135, COMP 136, or COMP 131 recommended but not required.

COMP 152-SBR Statstical Bioinformatics in R

The analysis and interpretation of multivariate biological data sets requires an understanding of statistical and computational methods and tools. This course will introduce students to underlying concepts and specific tools for working with several types of high-dimensional biological data using the R programming language. Topics include probabilistic distributions, statistical modeling, hypothesis testing, data visualization and cleaning, supervised and semi-supervised learning, network and graph representations of biomedical data. Applications include RNA-sequencing, metagenomics, phylogenetics, and disease informatics. Learning will be supported through regular computational homework assignments and a final project including computational and written components and a project presentation.

Please see the current course website at https://www.cs.tufts.edu/comp/152SBR/

Prerequisite: Comp 15 or equivalent, or graduate standing (with permission of instructor); some prior experience in R.

COMP 156-SEN Software Engineering

Software engineering is an engineered discipline in which the aim is the production of software products, delivered on time and within a set budget, that satisfies the client's needs. It covers all aspects of software production ranging from the early stage of product concept to design and implementation to post-delivery maintenance. This course covers the major concepts and techniques of software engineering including understanding system requirements, finding appropriate engineering compromises, effective methods of design, coding, and testing, team software development, and the application of engineering tools so that students can prepare for their future careers as software engineers. The course will combine a strong technical focus with a project providing the opportunity to obtain hands-on experiences on entire phases and workflow of the software process.

Prerequisite: COMP 40, graduate standing, or instructor consent.

COMP 177 Visualization

Visualization as a tool for data analysis, recall, inference, and decision-making. Tools for visual description and presentation. Principles of effective visualization, including data-visual mapping, interaction techniques, color theory, cognitive and perceptual psychology, and human factors of visual depictions of data.

Prerequisite: Comp15 and Comp61, or permission of instructor.

COMP 180 Software Engineering

Core principles and ideas that enable development of large-scale software systems, with a focus on programming. Abstraction, modularity, design patterns, specification, testing, verification, and debugging.

Prerequisite: Recommendations: COMP 40

COMP 181 Compilers

In COMP 181, you will learn about the design and implementation of modern compilers. The course will focus on the main steps of general compilation (Scanning, Parsing, Semantic Checking, and Code Generation), while also introducing some specific compilation techniques for language-specific features. Traditional compiler optimizations may also be introduced.

Outside the classroom, the focus of the course is an intensive, semester-long project: you and a team will design a small programming language and implement a compiler for it using the OCaml language. The algorithms and concepts you will learn have broad application outside of the course: many programming tasks can be understood as variations of interpretation or translation, and understanding how a compiler operates will further develop your abstract thinking skills and make you a better programmer.

Prerequisite: COMP 40, COMP 105, and COMP 170.

CS 1-CSC Collaborative Introduction to Computer Science

An optional preparatory course for students with no prior programming experience and limited experience in college-level STEM classes. Basics of programming including variables, control flow, subroutines, and problem solving in a hands-on, collaborative environment. The class will meet over the latter half of the semester, and prepare students to enter CS11 with prior exposure to topics that CS 11 discusses in depth. Pass/fail grading.

Prerequisite: Recommendations: high school algebra.

CS 4 Teaching Computer Science

This course will prepare undergraduates to function effectively and efficiently as undergraduate teaching assistants. Through this course, students will learn pedagogical techniques that match learner needs; discuss ethical and social concerns that UTAs face in the course of a semester; and problem solve together issues that arise as teaching assistants. This course is designed in a learner centered model requiring your active and engaged participation. Through your willingness to share your experiences and expertise and your collaboration with your fellow UTA we will together construct meaningful solutions to difficulty situations. Faculty from Computer Science will participate in some of the sessions as co-facilitators. Students will be expected to complete short readings; keep a reflective blog of your learning as a teacher; give a short final presentation on a topic of interest that you want to explore in more depth to help you in your TA class.

CS 5-IDH Introduction to Digital Humanities

The Humanities is increasingly making use of computerized methods in gathering and interpreting research data. Simultaneously, the world has seen a dramatic increase in the dissemination of numerical claims-—the products of data science--in forms ranging from infographics in newspapers to interactive visualizations on social media and other web sites. This course covers selected topics in computing approaches for the Humanities. The aim is to equip Humanities students with basic awareness and skills in using digital research methods and applying that knowledge to understanding and interrogating the numerical claims produced by data scientists. Additionally, the course aims to provide experience with a variety of computational tools to approach Humanities research 9 questions, topics, and datasets. All examples and the applications of the computational tools will be drawn from the Humanities. We hope to empower Humanities students to understand and interpret research conducted with computerized methods. After this course, they should be familiar with the steps involved in working with data. They will be prepared to ask questions of numerical claims they encounter in publications and online. Where did the source data come from? How was it processed? Why was it visualized in this way and how might the visualization help or hinder appropriate interpretation? They will also be prepared to engage in research together with data scientists and thus form better teams. Non-Humanities students will be

better equipped to engage with Humanists by getting a sense for the needs of the field. They will also gain awareness of the different topics explored in the Humanities (such as language, history, prosopography, geolocalization, etc). Students will develop an interest in these disciplines and add them to a well-rounded curriculum. Further, students will benefit from exposure to and hands on experience with real world applications of data science techniques in the Humanities. The format of the course will be modeled after introductory Computer Science (CS) courses at Tufts. The course is open to all Tufts undergraduates and will be counted as a Humanities credit.

CS 5-PCS Preparing for Career Success

This course is focused on helping Computer Science students prepare for finding roles in the technology industry. It covers topics such as strategic thinking about opportunities, skills assessment, marketing your skills, resume and marketing materials preparation, interview and practice, improving presentation skills, and career networking. Most class sessions will be virtual, with some class meetings in person.

Prerequisite: Students interested in registering for this class should fill out the form emailed to CS and DS students.

CS 10 Computer Science for All

Computers are indispensable tools for research. This does not only hold for more technical fields such as physics or chemistry but also for the Humanities and the Social Sciences. While most students are competent users of standard software such as word processing or spreadsheets, the real power of the computer is unleashed when we are able to program it ourselves and make it do exactly what we want it to do.

This course is aimed at people who want to learn how to use computer science to solve basic information processing problems, such as analyzing text data and performing elementary statistics on them. It will cover elementary principles of computer science and will teach the student to independently write their own programs in the computer language Python.

This course is meant for people who have little or no previous experience in computer science. Therefore, in this course we do not assume that the students already know how to write computer programs. However, computer programming is a skill, and learning a new skill takes substantial amounts of effort and time. So the fact that this course is aimed at beginners does *not* mean that it is easy, or that it will involve less work than our other introduction courses, like e.g. COMP 11. On the contrary, it is very likely the case that this course will involves *more* effort than other introductory programming courses, if only because the fact that we do not assume any previous experience means that the road to our goal is going to be longer.

IMPORTANT NOTE: Passing this course does NOT fulfill the A&S Mathematics distribution requirement.

CS 11 Introduction to Computer Science

The study of computer science centers on two complementary aspects of the discipline. First, computer science is fundamentally concerned with the problem-solving methodologies it derives from its foundational fields: the design principles of engineering, mathematical theory, and scientific empirical study. Second, these methodologies are applied in the complex context of a modern day computing system. In this course we will address both of these important aspects. As a means for developing your design skills, we will discuss the fundamental features of a high level, general purpose programming language -- namely C++-- and learn how to use it as a tool for problem solving. We will also consider the performance of solutions, and how to apply both analytical and empirical assessment techniques. Finally, we will explore the Unix operating system as a context for problem solving. (Additional 2 hr weekly lab time scheduled at first class meeting.)

Recommendations: High school algebra. No prior programming experience is necessary.

CS 12 Cyber for Future Policymakers

Relevance of computer technologies to policy development. Internet architecture and basic networking, the Web, cloud architectures, cryptography, security and privacy, AI and machine learning, and open-source systems. Developing technologies, including quantum computing and post-quantum cryptography.

Prerequisite: Recommendations: COMP 10 or COMP 11.

CS 13 How Systems Work

How computing systems work: bits, bytes, the representation of information, the CPU, assembly language, programming languages. Networking: including peering, packets, and the Internet. Algorithms and the fundamental limitations of computing.

Prerequisite: Recommendations: COMP 10 or COMP 11.

CS 14 Emerging Scholars in Computer Science

Weekly, peer-led workshops exploring topics in computer science. Emphasis on the collaborative and problem-solving nature of computer science. No prior programming experience is necessary. Students must apply to enroll – see departmental website for details.

Prerequisite: Prerequisite: first year or sophomore standing Corequisite: COMP 10 or COMP 11.

CS 15 Data Structures

A second course in computer science. Data structures and algorithms are studied through major programming projects in the C++ programming language. Topics include linked lists, trees, graphs, dynamic storage allocation, and recursion.

Prerequisite: COMP 11 or consent. This course and COMP 50-01 (COMP 50-PSS) may not both be taken for credit.

CS 20 Web Programming

An introduction to techniques, principles, and practices of writing computer programs for the World Wide Web. Server and browser capabilities and limits. Media types, handlers, and limitations. Web programming languages and techniques. Web security, privacy, and commerce. Lectures augmented with programming projects illustrating concepts and current practice.

Prerequisite: COMP 11; or COMP 10 and consent.

CS 21 Concurrent Programming

When we learn to program, we specify problem solutions as a single sequence of computations in a fixed, determined order. But the world isn't like that. Deer run into the woods, people talk on their phones, it rains. Nothing forces these things to happen one at a time, in a fixed order. They happen concurrently. We want to write concurrent programs, because we want to model the real world, because our computer systems actually have concurrent activities, and also to improve the performance or usability of our programs. The ubiquity of distributed applications and modern, multicore processors makes concurrent programming an essential skill.

This course explores different models of concurrent programming: students will gain competence in conventional shared-memory threads programming, and at least one natively concurrent programming model (actors or CSP). Time permitting, we may look at other models. We'll look at classic problems (like deadlock) and synchronization mechanisms (semaphores, locks, barriers).

Students will complete a substantial team programming project using these tools and techniques, and they will present their work to the class.

Prerequisite: CS 15, or graduate/postbac standing

CS 23 Game Design

Principles, design, and development of games. Game structure, engineering, physics, testing, 2D and 3D rendering, user interfaces, sound, and animation. Security of online games. Applications of Economics, Music, and Psychology in crafting games. Projects include writing game design documents, developing an interactive fiction game, and building a functional game in a team.

Prerequisite: Recommended: Comp 15.

CS 27 How Systems Fail

Failure of computer systems within the larger context of complex systems, including the power grid and aviation. Failures of algorithms and protocols, engineering and implementation, systems and applications, people and culture. Attacks, attack recovery, security, privacy, and attribution. Case studies of failures and attacks, including distributed denial of service, Meltdown, Spectre, and spear-phishing attacks.

Prerequisite: COMP 13 or consent of instructor.

CS 28 Cyber Security and Cyber Warfare

Interdisciplinary analysis of cybersecurity in the United States and other countries, intended to introduce engineering students to policymaking and intelligence aspects of cybersecurity and liberal arts students to the technical constraints of computer networks and software. Hands-on activities including packet analysis, exploiting a vulnerable system, password cracking, social engineering, reconnaissance, and malware analysis. Examination of state and non-state actors engaged in cyber-espionage, counterintelligence, deterrence, and offensive cyber operations. Guest speakers from private sector, civil liberties groups, and intelligence community.

Prerequisite: PS 61: Introduction to International Relations (for PS and IR majors) or COMP 15: Data Structures (for CS majors in A&S or SOE)

CS 30 Programming for Data Science

Fundamentals of programming for data-intensive science. Data structures and algorithms for data manipulation, cleaning, and preparation. Design of data manipulation programs. Coding standards and practices. Use and creation of software libraries. Techniques for improving program performance. Examples drawn from data preparation and transformation, statistical data analysis, machine learning, deep learning, and deep data science including recommendation systems and trend analysis.

CS 40 Machine Structure & Assembly-Language Programming

In COMP 40, you will learn about both high-level programming design principles and the low-level structure of computing machines. Design strategies will focus on modularity, abstraction, and separation of interface from implementation. The following topics on machine structure are covered: memory, caches, registers, machine arithmetic, and bitwise operations. We will also investigate the structure of assembly code, relocatable object code, binary machine code, and the translations between them. You will gain a deep understanding of all of these concepts via large-scale, realistic programming projects.

Mandatory lab will be held Fridays: sign up in SIS.

See https://engineering.tufts.edu/cs/current-students/undergraduate/high-demand-enrollment for the form required to get approval to enroll in this class.

Prerequisite: COMP 15.

CS 40 Machine Structure & Assembly-Language Programming

In COMP 40, you will learn about both high-level programming design principles and the low-level structure of computing machines. Design strategies will focus on modularity, abstraction, and separation of interface from implementation. The following topics on machine structure are covered: memory, caches, registers, machine arithmetic, and bitwise operations. We will also investigate the structure of assembly code, relocatable object code, binary machine code, and the translations between them. You will gain a deep understanding of all of these concepts via large-scale, realistic programming projects.

Mandatory lab will be held Fridays: sign up in SIS.

See https://engineering.tufts.edu/cs/current-students/undergraduate/high-demand-enrollment for the form required to get approval to enroll in this class.

Prerequisite: COMP 15.

CS 45 Computer Organization

Computer organization including performance measurement, instruction set architectures, digital arithmetic, processor datapath, control, pipelining, memory hierarchy, caches and input/output.

Prerequisite: Prerequisite: ES 4

CS 50-SDT Intro to Software Development Tooling

Effective software development requires more than just coding skill: in industry and academia alike, developers use tools to keep their code maintainable and reliable. In this course, you will learn four fundamental categories of tooling: version

control, the Linux shell, build systems, and testing. We'll dive deep into one industry-standard tool from each category via hands-on projects and exploration of existing codebases, then survey other tools in the same category and discuss why you might choose one over another. By the end of the course, you will have a robust toolset both to manage complexity in your future projects and to effectively ramp up on software projects you encounter in the real world.

CS 52-IDH Intro to Digital Humanities

Computational methods to study humanistic disciplines (literature, history, politics, art, music, social media, etc.). Introduction to methods such as web scraping, Application Programming Interfaces (APIs), topic modeling, Named Entity Recognition (NER), network analysis, mapping, and the programming language Python. Assignments analyze primarily text-based sources in English, and the methods can be applied to a wide range of languages, ancient and modern. No programming skills are assumed.

CS 52-NLP Natural Language Processing & Human Records

TBA

CS 61 Discrete Mathematics

(Cross-listed as Mathematics 61.) Sets, relations and functions, logic and methods of proof, combinatorics, graphs and digraphs.

Prerequisite: Math 32 or Computer Science 11 or permission of instructor.

CS 86 Object-Oriented Programming for Graphical User Interfaces (formerly Comp 106)

Object-oriented programming (OOP) and design, using the Java language. General OOP concepts (classes and instances, methods, inheritance) plus specifics of programming in Java, with emphasis on application to graphical user interfaces (GUIs). Design and programming projects using Java and toolkits.

Prerequisite: Comp 15 or permission of the instructor.

CS 93 Directed Study

Guided study of an approved topic. Credit as arranged.

Prerequisite: Consent.

CS 97 Senior Capstone Project I

Requirements analysis and design of a senior capstone project. Requirements analysis and elicitation methods, and prototyping. Design principles and methods, including designing for usability, security, testability, performance, and scaling. Project management and planning, including cost and effort estimation. Writing effective documentation.

Prerequisite: COMP40 and Senior Standing.

CS 98 Senior Capstone Project II

Implementation and testing of the project designed in CS 97. Implementation tools, strategies, and platforms. Testing and debugging methodologies. Maintenance and release management. Legal, ethical, and social impacts of computing.

Prerequisite: CS 97

CS 99 Internship Computer Science

Prerequisite: Consent.

CS 105 Programming Languages

Principles and application of computer programming languages. Emphasizes ideas and techniques most relevant to practitioners, but includes foundations crucial for intellectual rigor: abstract syntax, lambda calculus, type systems, dynamic semantics. Case studies, reinforced by programming exercises. Grounding sufficient to read professional literature.

Prerequisite: COMP 15 (Data Structures) and one semester of Discrete Mathematics (COMP/MATH 22 or 61).

CS 106 Virtual Machines & Language Translation

Translation of high-level, functional programming languages to virtual-machine code. Design and implementation of register-based virtual machines and instruction sets. Bytecode interpretation. Function calls including optimized tail calls. Virtual assembly language, virtual object code, and code loading. Automated memory management (garbage collection). Functional-programming techniques for translation. Composition of translation passes using an error monad. Parsing and unparsing. Parsing combinators. Translation of higher-order functions via closure conversion. Naive register allocation by reduction to K-normal form. Code generation from K-normal form to virtual assembly language. Lightweight benchmarking.

CS 107 Compilers

In COMP 181, you will learn about the design and implementation of modern compilers. The course will focus on the main steps of general compilation (Scanning, Parsing, Semantic Checking, and Code Generation), while also introducing some specific compilation techniques for language-specific features. Traditional compiler optimizations may also be introduced.

Outside the classroom, the focus of the course is an intensive, semester-long project: you and a team will design a small programming language and implement a compiler for it using the OCaml language. The algorithms and concepts you will learn have broad application outside of the course: many programming tasks can be understood as variations of interpretation or translation, and understanding how a compiler operates will further develop your abstract thinking skills and make you a better programmer.

Prerequisite: CS 40 and CS 105, or graduate standing, or permission of the instructor.

CS 111 Operating Systems

(Crosslisted as EE 128). Fundamental issues in operating system design. Concurrent processes: synchronization, sharing, deadlock, scheduling. Relevant hardware properties of uniprocessor and multiprocessor computer systems.

Prerequisite: Recommendations: COMP 15 and either CS 40 OR EE 14.

CS 112 Networks & Protocols

Design and implementation of computer communication networks, protocols, and applications, with an emphasis on the Internet protocol suite. Network architectures and programming interfaces. Data link, transport, and routing protocols. Congestion sources and remedies. Addressing and naming in local area and wide area networks. Network security and network management.

Prerequisite: Comp 15 and one of EE 14 or Comp 40

CS 113 Human Factors in Security and Privacy

Usability and human-computer interaction (HCI) problems of privacy and security. Common HCI methods that can be used to measure usability issues in security and privacy mechanisms. Practical experience understanding and designing studies which evaluate usability issues in security and privacy systems. Course instruction assumes students are familiar with introductory concepts in programing.

CS 114 Network Security

Vulnerabilities, attacks, and mitigations at all layers of the network stack. Public and private key cryptography, confidentiality and authentication protocols, botnets, firewalls, intrusion detection systems, and communication privacy and anonymity.

Prerequisite: Prerequisites: Computer Science 15 or graduate standing.

Recommendations: Computer Science 40.

CS 115 Database Systems

Fundamental concepts of database management systems. Topics include: data models (relational, object-oriented, and others); the SQL query language; implementation techniques of database management systems (storage and index structures, concurrency control, recovery, and query processing); management of unstructured and semistructured data; and scientific data collections.

Prerequisite: COMP 15.

CS 116 Introduction to Security

A systems perspective on host-based and network-based computer security. Current vulnerabilities and measures for protecting hosts and networks. Firewalls and intrusion detection systems. Principles illustrated through hands-on programming projects.

Prerequisite: Comp 15.

CS 117 Internet-Scale Distributed Systems: Lessons from the World Wide Web

Please note that this course was formerly numbered COMP 150-IDS.

The World Wide Web, one of the most important developments of our time, is a unique and in many ways innovative distributed system. This course will explore the design decisions that enabled the Web's success, and from those will derive important and sometimes surprising principles for the design of other modern distributed systems.

We will introduce and draw comparisons with more traditional distributed system designs, including distributed objects, client/server, pub/sub, reliable queuing, etc. We will also study a few (easily understood) research papers and some of the core specifications of the Web. Specific topics to be covered include: global uniform naming; location-independence; layering and leaky abstractions; end-to-end arguments and decentralized innovation; Metcalfe's law and network effects; extensibility and evolution of distributed systems; declarative vs. procedural languages; Postel's law; caching; and HTML/XML/JSON document-based computing vs. RPC.

The purpose of this course is not to teach Web site development, but rather to explore lessons in system design that can be applied to any complex software system.

More detailed course information can be found at https://www.cs.tufts.edu/comp/117/shouldItakeit

Prerequisite: CS 40 or permission of the instructor.

CS 118 Cloud Computing

Cloud computing fundamentals, including cloud architecture, scalability, elasticity, and metrics of cloud performance including service-level objectives (SLOs) and service-level agreements (SLAs). Cloud programming models and abstractions including Map/Reduce. Persistent storage mechanisms, including key/value stores and cold storage. Geo-distributed cloud systems. Cloud networking, including data center architecture, software defined networking, and middleboxes. Cloud security.

Prerequisite: COMP 15 and (COMP 40 or EE 14) or CS Grad standing or CS postbac

CS 119 Big Data

"Big Data" deals with techniques for collecting, processing, analyzing and acting on data at internet scale: unprecedented speed, scale, and complexity.

This course introduces the latest techniques and infrastructures developed for big data including parallel and distributed database systems, map-reduce infrastructures, scalable platforms for complex data types, stream processing systems, and cloud-based computing. The course content will be a blend of theory, algorithms and practical (hands on) work.

Prerequisite: A beginning course in databases, familiarity with Python, shell programming, Java, Scala, and SQL.

CS 120 Web Programming and Engineering

Web applications are complex systems that deliver a plethora of functionality to a large number of users, and also exhibit unique behaviors and demands in terms of performance, scalability, usability, and security. Web engineering is an emerging and multidisciplinary process that is used to create quality web applications. This course will discuss the limits of current web technologies, the similarities and differences between web and software engineering, design, information and service architectures, content management, and testing disciplines. Frameworks such as Rails, Spring, and Symfony will be emphasized and used. Projects will involve search, cloud computing, location-based services, and mobile web development.

Prerequisite: Comp 15 and Comp 20, or Consent of Instructor.

CS 121 Software Engineering

Software engineering is an engineered discipline in which the aim is the production of software products, delivered on time and within a set budget, that satisfies the client's needs. It covers all aspects of software production ranging from the early stage of product concept to design and implementation to post-delivery maintenance. This course covers the major concepts and techniques of software engineering including understanding system requirements, finding appropriate engineering compromises, effective methods of design, coding, and testing, team software development, and the application of engineering tools so that students can prepare for their future careers as software engineers. The course will combine a strong technical focus with a project providing the opportunity to obtain hands-on experiences on entire phases and workflow of the software process.

Prerequisite: COMP 40, graduate standing, or instructor consent.

CS 122 Parallel Computing

(Cross-listed w/ EE 155) Programming modern parallel computer architectures, especially GPUs and multi-core CPUs. Rationale for modern multi-core CPUs. Challenges of multi-threaded programming. High-performance software taking advantage of hardware caches, cache coherency, memory systems and parallel computation.

Prerequisite: Recommendations: EE 126 or COMP 40.

CS 125 Numerical Analysis

Analysis of algorithms involving computation with real numbers. Interpolation, methods for solving linear and nonlinear systems of equations, numerical integration, methods for ordinary differential equations.

Prerequisite: Recommendations: MATH 51 and programming ability in a language such as C, C++, Fortran, Pascal, or Matlab.

CS 126 Numerical Linear Algebra

(Cross-listed as MATH 126)The two basic computational problems of linear algebra: solution of linear systems and computation of eigenvalues and eigenvectors.

Prerequisite: Recommendations: MATH 70 or 72 and COMP 11.

CS 131 Artificial Intelligence

History, theory, and computational methods of artificial intelligence. Basic concepts include representation of knowledge and computational methods for reasoning. One or two application areas will be studied, to be selected from expert systems, robotics, computer vision, natural language understanding, and planning.

Prerequisite: Comp 15 and either COMP/MATH 22 or 61 or familiarity with both symbolic logic and basic probability theory.

CS 132 Computer Vision

Introduction to low and intermediate levels of classic and modern Computer Vision. How to design algorithms that process visual scenes to automatically extract information. Fundamental principles and important applications of computer vision, including image formation, processing, detection and matching features, image segmentation, and multiple views. Basics of machine learning and deep learning for computer vision.

Prerequisite: Recommendations: CS 160 and Math 165

CS 133 Human-Robot Interaction

This course will provide an overview of the up and coming field of human-robot interaction (HRI) which is located squarely in the intersection of psychology, human factors engineering, computer science, and robotics. HRI has become a major research focus recently with the NSF's National Robotics Initiative and the push countries around the globe to develop robots for various societal tasks, from new flexible and adaptive robots for industrial manufacturing, to socially assistive robots for eldercare. In this course, we will examine this field from an interdisciplinary perspective, reading key papers in HRI that intersect computer science, robotics, cognitive and social psychology (since there is no suitable textbook yet, all reading materials will be made available). Students will give short presentations on HRI studies and designs and work in interdisciplinary groups on a term project which will require them to design and conduct an HRI study.

Prerequisite: Senior or graduate standing in Computer Science, or permission of instructor.

CS 134 Computational Models in Cognitive Science

Scientific logic of using computational models for testing theories in cognitive science. Connectionist and Bayesian models; agent-based simulation. Emphasis upon using models in combination with empirical data to test theories. Appropriate use and critical evaluation of computational modeling as found in scientific publications. Recommendations: COMP 10, 11, or some programming experience.

CS 135 Introduction to Machine Learning

An overview of methods whereby computers can learn from data or experience and make decisions accordingly. Topics include supervised learning, unsupervised learning, reinforcement learning, and knowledge extraction from large databases with applications to science, engineering, and medicine.

Prerequisite: Comp 15 and COMP/MATH 22 or 61 or consent of instructor. (Comp 160 is highly recommended).

CS 136 Statistical Pattern Recognition

Statistical foundations and algorithms for machine learning with a focus on Bayesian modeling. Topics include: classification and regression problems, regularization, model selection, kernel methods, support vector machines, Gaussian processes, Graphical models.

Prerequisite: Prerequisites: MATH 13 or 42; MATH 46 or 70; EE 104 or MATH 166; CS 40 or CS 105 or a programming course using Matlab. CS 135, or CS 131 are recommended but not required. Or permission of instructor.

CS 137 Deep Neural Networks

Deep neural networks and their applications including computer vision and natural language processing. Feed-forward, convolutional, and recurrent neural networks. Techniques for training deep neural networks, including optimization, regularization, and usage of related software.

Prerequisite: Recommendations: COMP 135, MATH 42, and MATH 70; or consent of instructor.

CS 138 Reinforcement Learning

"Reinforcement learning problems involve learning what to do — how to map situations to actions — so as to maximize a numerical reward signal." - Sutton and Barto ("Reinforcement Learning: An Introduction", course textbook)

This course will focus on agents that much learn, plan, and act in complex, non-deterministic environments. We will cover the main theory and approaches of Reinforcement Learning (RL), along with common software libraries and packages used to implement and test RL algorithms. The course is a graduate seminar with assigned readings and discussions. The content of the course will be guided in part by the interests of the students. It will cover at least the first several chapters of the course textbook. Beyond that, we will move to more advanced and recent readings from the field (e.g., transfer learning and deep RL) with an aim towards focusing on the practical successes and challenges relating to reinforcement learning.

There will be a programming component to the course in the form of a few short assignments and a final projects.

Approved as a category 2 elective in Data Science (analysis and interfaces).

Prerequisite: Students are expected to be proficient programmers in at least one of the following languages: C++, Java, or Python. Prior coursework (or experience) in Artificial Intelligence and/or Machine Learning is highly recommended, but not required.

CS 139 Ethics for AI, Robotics, and Human Robot Interaction

This course will provide an overview of the ethical problems and challenges prompted by current and future technological advances in AI, robotics, and human-robot interaction. It will start by reviewing the philosophical foundations of the main ethical theories (virtue ethics, deontology, utilitarianism) and link them to different algorithmic approaches in artificial agents (rule-based, utility-based, behavior- based, etc.). Explicating and contrasting the assumptions underlying each algorithmic approach (e.g., policy-based decision-making vs. rule-based reasoning), functional tradeoffs and implications for autonomous robots and AI systems will be discussed. The scope will then be widened to moral psychology and human-robot/human-technology interaction to move beyond individual autonomous systems into the realm of social interactions between humans and autonomous systems, discussing the societal implications of AI and robot technology. Social, economical, legal, and military ramifications will be considered, with the aim of exposing the unique challenges AI and robot technology pose for humanity, compared to other disruptive technologies, but also they unique opportunities these technologies enable for current and future generations.

Prerequisite: Senior standing, or permission of instructor. Recommendations: CS/MATH 61 and CS 15. This is an undergraduate-only course; graduate students in the HRI program will need to enroll in CS 239-01.

CS 140 Advanced Topics Computer Architecture

(Cross-listed w/ EE 156) Modern computer architecture, starting from basic 5-stage pipelines and progressing to out-of-order superscalar processors, multicore processors, and heterogeneous processors. Techniques to maximize single-thread performance within the constraints of memory technology, power consumption, and the inherent instruction-level parallelism of applications. Current and future challenges faced by computer architects and computer-system designers. Discussion of research papers.

Prerequisite: Recommendation: EE126/COMP146 or COMP40

CS 141 Probabilistic Robotics

Techniques for probabilistic state estimation and their application to problems such as robot localization, mapping, perception, and planning in the context of Human-Robot Interaction. Machine learning and computer vision techniques commonly used by robots interacting with humans. Recommendations: Proficiency in C/C++ or Python

CS 142 Network Science

Mathematical foundations of the study of graphs and networks that arise as social, biological and Internet networks. Random graph models, community structure and inference problems, network dynamics, cascading. Example networks drawn from the application domains will be case studies as a companion to the general mathematical theory.

Approved as a category 2 elective in Data Science (analysis and interfaces).

Prerequisite: Recommendations: MATH 70 or 72 or CS 135 or 160

CS 144 Iterative Methods in Machine Learning

(Cross-listed as EE 143) Design and analysis of modern machine learning methods with emphasis on convex and nonconvex problems, and centralized, federated, and distributed computational architectures. Topics include convergence, complexities, contractions, fixed point theorems, and perturbation techniques; gradient descent and stochastic gradient descent in addition to accelerated methods including Polyak and Nesterov momentum, minibatching, and variance reduction. State of the practice methods will be covered including Adagrad, Autogard, Adam, sgdm with applications in image classification and document clustering.

Prerequisite: MATH 70 and CS 11

CS 146 Computer Engineering with Lab

(Cross-listed w/ EE 126.) This course teaches advanced concepts of modern computer architecture, starting from the basic 5-stage pipelines and progressing to out-of-order superscalar processors. This course introduces the techniques used to maximize single-thread performance within the constraints of memory technology, power consumption, and the

inherent instruction-level parallelism of applications. Students will also gain hands on hardware experience by implementing a 5-stage MIPS processor in VHDL. Recommendations: EE 14 or COMP 40; and ES4.

Prerequisite: ES4 and either EE14 or COMP 40 with a 'C' or better

This course assumes that undergraduate students have taken an undergraduate-level introduction to assembly programming (EE14 or COMP40) and are also proficient in digital logic design (ES4 recommended). In addition, this course (EE 126/COMP 46) will be required to take Advanced Computer Architecture (EE 156 and COMP 140) which is typically offered in the spring semester.

CS 149 Information Theory

Information theory as a systematic framework to address fundamental laws and limits of data compression and digital communication. Source coding/data compression; information measures on discrete memory-less sources; practical schemes and algorithms for lossless data compression such as Huffman coding, arithmetic coding, Lempel-Ziv Coding; channel coding for reliable communication and rate distortion for lossy source compression. Advanced topics such as information theoretic cryptography.

Prerequisite: Recommendations: Undergraduate Probability OR EE 104 OR Permission of instructor.

CS 150-AEP Artificial Intelligence: Algorithms, Ethics, and Policy

Artificial intelligence (AI) has emerged as a transformative technology that is being adopted in a wide range of settings with significant impact on society. This course will examine the ethics of AI usage in these settings and how policy can affect this usage. The course will also introduce technical aspects of AI algorithms. With combined perspectives on policy and technical knowledge, students who complete the course will be equipped to become informed policy experts and decision-makers with substantial understanding of the technology necessary to lead on issues involving AI. This new course is still under development, so details are TBA and subject to change.

Note: Cross listed with DS153-01.

CS 150-AISI Generative AI for Social Impact

Prerequisite: Completion of CS 15 or graduate standing.

CS 150-AMC Algorithmic Music Composition

CS 150-CAFM Cognitive Architectures Age of Foundation Models

This course will delve into the changes to cognitive architectures prompted by the rise and success of foundation models, in particular, large language models. Part of the goal of cognitive architecture was to capture human cognition and intelligence. Arguably, foundation models have made more progress on that end, at least on the intelligence part, than any of the past cognitive architecture were able to achieve. Yet, foundation models are not cognitively or neurally plausible, and they have a host of problems of their own. The course will review classical cognitive architecture and discuss ways for utilizing foundation models to improve them, e.g., by replace components of classical cognitive architecture with foundation models. The course will provide opportunities for students to computationally investigate different integration methods and their design tradeoffs in embodied and disembodied computational models.

Prerequisite: Solid programming knowledge in Java and Python and willingness to learn new representations and programming paradigms (e.g., production systems in classical cognitive architectures), knowledge in AI, robotics, algorithms, data structures, statistical and experimental methods.

Open to CS grad students and advanced CS UGs and CBS students with excellent programming skills and the ability to work independently on programming assignments

CS 150-CRY Cryptography

This is an introduction to cryptography, starting with the first ciphers, and leading up to present day issues. We will discuss how codes and ciphers work, and how they can be broken. We will cover both Private key (Symmetric) and Public Key (Asymmetric) cryptography. Topics include: cryptographic protocols using block ciphers. Methods for key exchange, hashing, message authentication, and digital signatures. Cryptographic protocols regarding secret sharing and digital cash. We will use a mathematical approach to prove properties about the crypto systems we study.

Prerequisite: Requirements: completion of COMP 15 and COMP/MATH 61, or grad standing. Recommendations: Comp 160, or Comp 170, or any 100 level Math course

CS 150-CTF MITRE eCTF

MITRE's Capture the Flag challenge

CS 150-DCC Debugging Cloud Computing

Cloud computing, which is the practice of renting software and hardware services from providers who run large-scale data centers, has become critical to modern society. We rely on software running within cloud data centers when shopping (e.g., at Amazon), when conducting financial transactions (e.g., at an online broker), when collaborating at work (e.g., using Google Docs), and even when playing games (e.g., Fortnite). Failures or performance problems within these data centers or the software running on them can have widespread effects and be devastating.

In this course, we will examine failures in cloud environments and discuss important research on tools that use systems knowledge, machine learning, and statistics to help engineers diagnose them. To provide students with necessary background, we will start with a brief introduction to cloud computing and the software systems that make cloud computing possible. The course will involve reading research papers, homework assignments, and coding-based projects. It is recommended for graduate students and advanced upper-level undergraduates.

Prerequisite: CS 15 and CS 40 or graduate standing required; CS 111 recommended.

CS 150-DGL Deep Graph Learning

Graph and network data are ubiquitous and often have a large scale. Graph data are generally characterized by the graph structure and data attached to graph nodes or edges. Machine learning is an important approach to automated information extraction from graph data. However, graph data need special model designs, as most learning models (e.g., neural networks) only accept vectors as the input. In this course, we will introduce a list of models that can extract information from graph data. Furthermore, we will also discuss the principles behind these models from the perspective of graph theory. The coursework consists of quizzes, 3 projects, and a final project.

CS 150-DPS Data and Power: Surveillance

What does it mean to do data science in a society filled with competing economic, social, and political interests? In this course, we will draw from historical perspectives on present-day issues in computer science to develop a social and political understanding of what the problem is, how it came to be, and what we can do about it. Our topic of focus will be surveillance, a problem that intersects economics through capitalism, race through histories of the policing of Black and Brown people, and power through systems of control. Throughout the three arcs comprising the class – (1) defining surveillance and learning its history, (2) developing an ethical inquiry and critique of surveillance technology, and (3) applying this practice to present day related problems – we will build “muscles” of analysis and practice that will build a foundation for budding data scientists to navigate a social and political world.

CS 150-ECS Entrepreneurship for Computer Scientists

(Cross listed with ELS 194-02) 150 ECS is an introductory entrepreneurship course for Computer Science students. The course provides an overview of entrepreneurship, develops an entrepreneurial perspective, and provides a framework for learning the fundamentals of the essential elements of entrepreneurial ventures, specifically directed toward software-related industries and products. Students learn how to develop their technical ideas into potential business opportunities, and to explore their likelihood of becoming viable businesses. They learn how to do market research, to develop go-to-market strategies, value propositions and to differentiate their products or services from actual or potential competitors. The course consists of a balance of lectures, projects, case studies and interaction with entrepreneurs and computer scientists who participate in entrepreneurial organizations.

CS 150-EP Epistemic Planning

This course is intended for students of computer science, mathematics, philosophy, or similar fields who are interested in automated logic-based deliberation. Course work will involve reading research papers, written assignments, and implementing computational techniques. Participants should know how to program.

Automated task planning involves determining what actions should be performed in order to achieve some goal within a task environment. Epistemic Planning involves not only objective facts about the environment, but also the beliefs and knowledge of agents within that environment. Instead of simply trying to determine "how do I get to the cafeteria?", one

considers problems such as "how do I determine which cafeteria has pizza, and then get to that cafeteria, while also making sure that my friend knows where I'm going"?

Epistemic Planning has emerged mainly from the intersection of three fields: Dynamic Epistemic Logic, Knowledge Representation and Reasoning, and Automated Planning. We will develop a foundational background in these topics and then study the development of Epistemic Planning techniques over the last approximately ten years.

CS 150-ETH Ethical Issues in Computer Science and Technology

A study of ethical issues connected with computer science and computing technology. We will read in the original philosophical literature of ethics to examine various theories of what is or is not ethical, and look at justifications for ethical behavior. These readings will be paired with case studies examining ethical or non-ethical uses of computing technology.

CS 150-GT Graph Theory

Introduction to graph theory, including trees, matchings, coloring, planar graphs, random graphs, algebraic graph theory. This class does not require programming, but there are applications in computational geometry, computational biology, and algorithms. Students may select either an implementation or a theoretical final project.

Prerequisite: CS 61 Discrete Math with a grade of B+ or better OR [any one of: COMP 160, COMP 170, any math numbered 100 or above] OR graduate standing

CS 150-HFD HCI for Disability

This is a graduate-level course for research-oriented students (including seniors and MS students who are considering research careers). Through readings, discussion, and a substantial course project we will explore the variety of ways that the human-computer interaction (HCI) research community has addressed the needs of disabled users, both from the perspective of assistive technology, which develops tech to support disabled people in their daily lives, and from the perspective of accessibility, which considers how disabled people can access computers and computing technologies. We will read and discuss academic papers from the HCI community (especially papers from ASSETS and CHI) in parallel with personal narratives from disabled people themselves and readings from the disability studies literature.

By the end of the course, students will:

- be familiar with core themes of disability studies and the disability justice movement as they relate to HCI.
- understand key considerations of HCI research with and for disabled people.
- be familiar with a variety of papers in HCI relating to accessibility and assistive technology across a wide range of intended user populations.
- learn to engage critically with the HCI literature from both a technical and disability studies perspective.
- complete an HCI research project relating to accessibility or assistive technology.

The final grade will be based on performance on the project and project milestones, as well as discussion of readings, including leading discussion. Participation in most class discussions and substantial amounts of reading and writing are key components of this iteration of the class.

CS 150-LAI Logic for AI (or CS + AI)

The course will provide an overview of different logical systems to students interested in applications in AI, robotics, machine learning, programming languages, natural language understanding, and the semantic web. Starting with a brief review of standard propositional and first-order logic, the course will quickly dive into different modal logics, their proof systems, semantics, and model checking algorithms which are required for various applications in different CS fields. Of particular interest will be epistemic and temporal logics, but there will also be opportunities to discuss other non-standard logics that have applications in different areas of CS based on student interests.

Prerequisite: Senior standing or grad student

CS 150-LLM Large Language Models

CS 150-MAS Multi-Agent Systems

Prerequisite: Prior completion of CS 131

CS 150-PLD Programming Language Design

Creating a new programming language requires designers to draw on a wide range of skills: theory to ensure their creation has a well-defined semantics, engineering to ensure they produce an efficient implementation, and aesthetics to ensure the whole is coherent and pleasing to programmers. In this class, we will study the principles and tools that underlie programming language design. We will consider both general-purpose languages, which are languages intended for almost any programming task, and domain-specific languages, which are languages focused on specific tasks. The class will be a combination of lectures and seminar-style discussions about selected academic papers. Students will use the knowledge they acquire to design and implement their own domain-specific programming language.

CS 150-QCS Quantum Computer Science

This is an elementary-level introduction to quantum computing through the computer science lens. One of the goals of this course is to present a language of quantum mechanics that is accessible to students working in computer science and vice versa. Topics include Hilbert spaces, the Dirac notation, the Schrödinger equation, quantum circuits, quantum Fourier transforms, quantum algorithms, and physical realizations of quantum computers. Students from different areas of engineering and sciences such as computer science, physics, electrical engineering, mathematics, or chemistry who wish to learn about the computer science foundations of quantum computers can benefit from this class.

Prerequisite: Linear algebra MATH 70 (Linear Algebra) or equivalent. Basic familiarity with discrete math, algorithms, and calculus is also recommended.

CS 150-QCT Quantum Complexity Theory

This course is an introduction to the complexity theoretic foundations of quantum computing and centers around the question: “what classes of computational problems are (not) solvable on computational models with quantum mechanical information processing resources?”. We start off with a gentle introduction to quantum mechanics, quantum computing, models of computation and computational complexity. We then proceed to defining quantum complexity classes and their relationship to the complexity zoo. We then move to special topics such as Hamiltonian complexity, noisy intermediate-scale quantum complexity theory, quantum interactive proof systems, bosonic quantum complexity classes, the interface between physics and theoretical computer science, and more, depending on interest. The goal is to bring students to the research frontier.

Prerequisite: Completion of CS 170 and Introduction to Quantum Information Science or equivalent or permission of instructor.

CS 150-QIS Quantum Information Science

This is a graduate seminar focusing on special topics in quantum computing and information science. The primary goal of this seminar is to prepare enthusiastic students to explore the literature and perform research in this field. Another goal is to work together as a class and prepare pedagogical materials that explain basic concepts in quantum computing and information science using simple and accessible language. We will post these materials as blog posts as we cover different topics.

Depending on student interest, potential topics include, but are not limited to, algorithms, post-quantum cryptography, quantum complexity classes, Hamiltonian complexity, nonlocal games, applications of quantum information science in other areas of physics, mathematics, and computer science, such as quantum-inspired algorithms, machine learning, satisfiability, phase transitions, probabilistically checkable proofs, connections with quantum gravity, etc. There are no assignments or exams. Each student is responsible for contributing to class discussions, presenting special topics, and leading discussion sessions. They are furthermore responsible for writing reports explaining the concepts covered in these sessions using simple language and collectively preparing them as blog posts. We will make these blog posts available to the public. Enthusiastic undergraduate students are welcome to join this class. Students from CS, mathematics, and physics can benefit from this class.

Prerequisite: Quantum complexity theory (CS-150), quantum computer science (CS-150), Quantum Information and Quantum Computation (COMP 151-02), equivalent courses, or permission from the instructor. Strong background in related areas of physics and mathematics is recommended. Strong background in linear algebra is specifically required (e.g. linear operators and their properties). Solid background in areas of computer science, such as the theory of computing, algorithms, and complexity theory will be very helpful.

CS 150-QSD Quantum Software Development

Prerequisite: MATH 70 and any one of EE 24, EE 104, EE 140, MATH 165, MATH 166, or CS 136

CS 150-SUR Surveillance

What does it mean to do data science in a society filled with competing economic, social, and political interests? In this course, we will draw from historical perspectives on present-day issues in computer science to develop a social and political understanding of what the problem is, how it came to be, and what we can do about it. Our topic of focus will be surveillance, a problem that intersects economics through capitalism, race through histories of the policing of Black and Brown people, and power through systems of control. Throughout the three arcs comprising the class – (1) defining surveillance and learning its history, (2) developing an ethical inquiry and critique of surveillance technology, and (3) applying this practice to present day related problems – we will build “muscles” of analysis and practice that will build a foundation for budding data scientists to navigate a social and political world.

CS 150-TCC Topics in Computational Complexity

In this course we will study the computational complexity in concrete computational models. The main focus will be on various techniques to prove lower and upper bounds on the computational complexity in these models. We will mostly consider decision trees, communication complexity and Boolean circuit complexity. We will also briefly discuss proof complexity and algebraic computation models.

Prerequisite: CS 61 and a course on probability (MATH 165 or EE 104) are required to take this class. MATH 70 and CS 170 are recommended. CS 160 would be helpful.

CS 150-TCT Theoretical CompSci Toolkit

CS 150-TCT Theoretical CompSci Toolkit

In this course, we will explore various methods and techniques that are useful in various areas of computation theory. We will see how combinatorial and probabilistic technique, polynomial approach, Fourier analysis and linear algebraic technique can be applied to solve various problems in theoretical computer science. We will illustrate the techniques by the examples in theory of algorithms, computational complexity, learning theory, property testing and other areas.

Prerequisite: Prior completion of CS 61 or graduate standing.

Recommendations: CS 170 or equivalent is highly recommended, but is not required. CS 160 or equivalent would be helpful, but is not necessary. MATH 70 (Linear algebra) is recommended to have prior to this course, but this is not a prerequisite.

CS 150-THR Trust in Human-Robot Interaction

This class takes a cognitive science-focused look at how humans trust robots. We'll start by defining trust as it relates to robots (and compared to humans or other technology), and throughout the semester cover topics such as: what factors affect different types of trust, how do we measure trust both subjectively and behaviorally, what may happen if a robot is over trusted or under trusted, how can we model trust computationally, and what ethical issues can be tangled up with trying to build a trustworthy system. This will be a primarily discussion-based class, and the main class project will be to come up with a research project about trust in human-robot interaction and implement it in a human-robot interaction system.

Prerequisite: Completion of CS 133 recommended

CS 151-ACT Anonymous Communications Theory

We know how to communicate a message so that only the recipient can read the message. (We can just encrypt the message.) But how can we communicate over the Internet without anyone learning *who* we are communicating with? Anonymous channels can help the Iranian protester who wishes to inform the world what is happening in the streets of Tehran by tweeting videos and the netizen in Moscow who wants to read the BBC news (currently banned in Russia). We could use Tor (i.e., “The onion router,” inspired by Chaum's onion routing idea) or VPN, but both are easily blocked, and neither guarantees privacy from the adversary with full view of the network traffic (e.g., a standard model for a resourceful ISP- or AS-level adversary). In this course, we present cryptography-style definitions of anonymity and study state-of-the-art techniques for achieving provable anonymity.

Prerequisite: Prerequisites: Math 21 and CS 170 or graduate standing.

CS 151-CTV Addressing Cyber Threats, Vulnerabilities

Computer and information security can be understood as a discipline dealing with risk to computer and information systems and the data they process. This class will cover the analysis, assessment, understanding and management of risk and its components (threat, threat actor, vulnerability, impact, likelihood) from a technical perspective as well as a managerial one: • Understanding the concepts of threat, threat actor, vulnerability, likelihood and impact in the context of risk. • Key technical and non-technical threats, vulnerabilities and risks:

• Hardware vulnerabilities (e.g. Spectre & Meltdown) and mitigations • Software vulnerabilities (OWASP Top 10) and mitigations • Network and Architecture vulnerabilities and mitigations • Threats, vulnerabilities, and risks related to physical factors, human factors, and human-computer interaction • The real world: where resources are limited and all the above factors interact

• Risk in an organizational context • Quantitative and qualitative methodologies (e.g. FAIR, threat modeling) to collect data on, and understand, risk and its components. • How to decide what (and how) to fix or address vulnerabilities, threats, and risks • How to manage threats, vulnerabilities, and risks. How to deal with risks, e.g. how to mitigate technical issues.

Prerequisite: CS 201, CS 203, CS 15, or CS graduate standing

CS 151-CYC Cybersecurity Clinic

Students are placed in interdisciplinary teams to work on a cybersecurity project with non-profit organizations in the Medford, Somerville, Cambridge, and Greater Boston communities.

Prerequisite: Department consent required.

CS 151-DCC Debugging Cloud Computing

Cloud computing, which is the practice of renting software and hardware services from providers who run large-scale data centers, has become critical to modern society. We rely on software running within cloud data centers when shopping (e.g., at Amazon), when conducting financial transactions (e.g., at an online broker), when collaborating at work (e.g., using Google Docs), and even when playing games (e.g., Fortnite). Failures or performance problems within these data centers or the software running on them can have widespread effects and be devastating.

In this course, we will examine failures in cloud environments and discuss important research on tools that use systems knowledge, machine learning, and statistics to help engineers diagnose them. To provide students with necessary background, we will start with a brief introduction to cloud computing and the software systems that make cloud computing possible. The course will involve reading research papers, homework assignments, and coding-based projects. It is recommended for graduate students and advanced upper-level undergraduates.

Prerequisite: CS 15 and CS 40 or graduate standing required; CS 111 recommended.

CS 151-IMA Introduction to Mobile Application Development on iOS and Swift

This course introduces the basics of contemporary mobile application development using Apple's iOS platform. The main requirement of the course is to build a functioning application in iOS. The course is divided into six modules, each of which covers a different aspect of development which is used in a final project. Module 1 begins with the major features of the Swift programming language and its standard library, along with use of the Xcode IDE for Swift development. Basic language features are covered lightly so that extensive discussion may be focused on differentiating features of the language including closures, optionals, the Swift type system (tuple/enum/struct/class/func), and generics. Module 2 focuses on elements of Functional Reactive Programming with Apple’s Combine library. Module 3 deals with correct application architecture, using a uni-directional dataflow model. Module 4 covers drawing, touch handling, layout and programming for devices of various sizes and aspect ratios, making extensive use of Apple's SwiftUI technology. Module 5 takes the student through a full animation cycle showing which elements of the UI can be smoothly animated and how. Module 6 finishes instruction with a discussion of navigation, tabular data presentation, and application state management. Frequent small assignments progress from basic programming to realistic app development with a focus on responsive device graphics and algorithms. Code design and architecture are emphasized. The course culminates in a final project which integrates all topics discussed into a single coherent whole.

Prerequisite: Completion of CS 15 or graduate standing.

CS 151-IQI Intro to Quantum Information

TBA

CS 151-OFS Offensive Security

A hands-on, in-depth approach to building cyber capabilities on Windows 10 (x64). We will cover Windows architecture, user-mode execution, OS primitives, PE file format, process injection, shellcode, and building extendable/modular software. This course includes a strong technical focus using hands-on learning and pair programming; as we complete a cumulative project. We will also cover the ethics of developing offensive capabilities and why understanding how to build cyber capabilities will prepare students for a career in the security industry.

Prerequisite: CS 40 or graduate standing

CS 151-PHPC Parallel and High-Performance Computing

This course is structured to train students to reason about the design and implementation of efficient parallel programs. The focus areas of the class will be on the modeling, implementation and evaluation of distributed, message-passing interface (MPI) based programs, shared-memory thread-based OpenMP programs, and hybrid (MPI/OpenMP) programs. Almost all examples will be aligned with numerical scientific computing. This course is appropriate for those that want to transform serial algorithms into parallel algorithms, want to modify currently existing parallel algorithms, or want to write parallel algorithms from scratch. The assignments/projects will be parallelized using OpenMP and MPI (Message Passing Interface). Students should have a basic understanding of partial differential equations, linear algebra, sequential algorithms, and data structures, as well as reasonable programming skills (preferably C/C++ or Fortran) and familiarity with Linux or Unix.

Prerequisite: CS 15, MATH 42, and MATH 70 or graduate standing

CS 151-PIC Principles of Internet Comm.

Design and implementation of computer communication networks, protocols, and applications, with an emphasis on the Internet protocol suite. Network architectures and programming interfaces. Data link, transport, and routing protocols. Congestion sources and remedies. Addressing and naming in local area and wide area networks. Network security and network management. This course is the same as networking (CS 112) except the programming assignments are replaced by written homework and a research paper.

CS 151-PSD Privacy, Security, and Data

Organizations today collect and analyze massive amounts of information for important decision-making applications. However, these large-scale analytics often compromise sensitive user information, such as medical or financial records. In this course, we will survey and apply state of the art techniques in privacy-preserving data science, such as differential privacy and secure multi-party computation, to learn how to build systems that provide useful results while still respecting user privacy. Students will be expected to read research papers, give in-class presentations, and complete a final project utilizing real-world data.

Prerequisite: CS 115-level knowledge; some experience with Python, C++, and basic probability.

CS 151-QCS Quantum Computer Science

The universe at the sub-atomic scale is governed by quantum mechanical laws, which fundamentally differ from classical laws of motion. What is the nature of computation in such scales? Can we use quantum mechanical particles to perform computations? These are the core questions of the field of quantum computing. In this course, we present an elementary-level introduction to the computer science foundations of quantum computing. Topics include Hilbert spaces, quantum entanglement, quantum measurements, quantum circuits, quantum protocols and algorithms, Hamiltonians and the ground state problem, and quantum error-correcting codes. Students from different areas of engineering and sciences, such as computer science, physics, electrical engineering, mathematics, or chemistry, who wish to learn about the computer science foundations of quantum computers can benefit from this class. The main focus of this course is on the theoretical foundations of quantum computing; mathematical enthusiasm and knowledge in areas such as linear algebra, algorithms, discrete mathematics, and calculus are required.

Prerequisite: Math 34 and Math 70 and one of (CS 61 or Math 61 or Math 65)

CS 151-SC Sustainable Computing

Data centers require enormous amounts of electricity to operate and cool, so understanding sustainable computing practices for improving energy efficiency can significantly reduce environmental impact. This course delves into cutting-edge research in system-level power, energy, and thermal management. We will explore techniques at the operating system, network, and application levels to optimize power consumption and thermal efficiency while maintaining performance and reliability.

Prerequisite: CS 40 or Graduate Standing

CS 151-SPPT Security, Privacy, People, and Technology

Our lives are intertwined with computers, which makes their security and trustworthiness ever more important. While the former is necessary for the latter, more than security is needed to make a system trustworthy. Thus, in this class, we will learn about security technologies but also discuss architectural, organizational, and social factors as they relate to building trustworthy systems. We will explore the limits of computers and technology, and where "human" mechanisms must step in, and where in turn, these measures might be lacking

Prerequisite: CS 201, CS 203, CS 15, or graduate standing.

CS 152-AEP AI: Algorithms, Ethics, Policy

Artificial intelligence (AI) has emerged as a transformative technology that is being adopted in a wide range of settings with significant impact on society. This course will examine the ethics of AI usage in these settings and how policy can affect this usage. The course will also introduce technical aspects of AI algorithms. With combined perspectives on policy and technical knowledge, students who complete the course will be equipped to become informed policy experts and decision-makers with substantial understanding of the technology necessary to lead on issues involving AI. This new course is still under development, so details are TBA and subject to change.

CS 152-BDL Bayesian Deep Learning

The emerging research area of Bayesian Deep Learning seeks to combine the benefits of modern deep learning (scalable gradient-based training of flexible neural networks for regression and classification) with the benefits of modern Bayesian statistical methods to estimate probabilities and make decisions under uncertainty. The goal of this course is to bring students to the forefront of knowledge in this area through coding exercises, student-led discussion of recent literature, and an in-depth project. Covered topics include key modeling innovations (e.g. function approximation and deep generative models), learning paradigms (e.g. variational inference), and implementation using modern automatic differentiation frameworks. By completing a 2-month self-designed research project, students will gain experience with designing, implementing, and evaluating new contributions in this exciting research space.

Prerequisite: Completion of EE 104 or CS 136 or equivalent required

CS 152-CVI Computer Vision

This course is an introduction to low and intermediate level Computer Vision. We will learn how to design algorithms that process visual scenes to automatically extract information. The course will cover fundamental principles and important applications of computer vision, including image formation, processing, detection and matching features, image segmentation, and multiple views.

CS 152-HFS Human Factors in Security and Privacy

Humans are often viewed as the weakest link in security. However, there is growing recognition that technology alone is insufficient to solve all security and privacy problems. Human factors play and essential role. A provably secure system is only as secure as the way users choose to use it, and system builders need to account for these user decisions if they wish to provide security and usability. In this class, we will cover a variety of usability and human interaction (HCI) problems of privacy and security. We will also cover common HCI methods that can be used to measure usability issues in security and privacy mechanisms. Students are expected to complete homeworks on the topic and complete a semester-long researh project designed to give students practical experience understanding and designing studies which evaluate usability issues in security and privacy systems.

CS 152-HIS Health Information Systems

Description: Health Information Systems provide technology and enable information exchange for healthcare enterprises, health information exchanges, health insurers and other participants in the healthcare industry. This is a rapidly changing and evolving field, laying the foundation for improvements in healthcare efficiency, quality, and health outcomes. The course provides an overview of key healthcare information technologies and concepts: healthcare data and analytics, electronic health records (EHR), health information exchanges (HIE), healthcare information privacy and security, telemedicine, consumer health and mobile health systems, and population health management. A number of case studies provide additional analysis of technology challenges and solutions in healthcare informatics.

CS 152-LLLD Learning from Limited Labeled Data

This course will study machine learning methods that can learn from available labeled datasets of limited size to perform a desired task well by leveraging either some other abundant source of related data or a pre-trained model. Topics will include self-supervised learning, semi-supervised learning, transfer learning, and more. The goal of this course is to bring students to the forefront of knowledge in this area. Students will engage in discussions of recent literature and complete a semester-long team research project designing, implementing, evaluating, and communicating new contributions to this research area.

Prerequisite: Required: CS 135. Recommended: CS 137.

CS 152-PGR Plan and Goal Recognition

This course introduces concepts and techniques for recognizing behavior, intentions, goals, and plans. It presents formal definitions for various recognition problems, as well as a set of algorithms and approaches to solve each of them. The course consists of critical reading of papers, presentation of a paper in front of peers, and group discussions.

Prerequisite: CS 131 or graduate standing

CS 152-SAN Sports Analytics

This course plots the various intersection points between Sports Analytics and Computer Science, including pit stops in machine learning, computer vision, and computational geometry among others. Students will get hands-on experience working with real data and real problems for which there may or may not be absolute right or wrong answers. Assessments will be based on the students' ability to correctly implement the empirical strategies discussed in class, clearly explain their methods, and creatively transform raw data into actionable insights.

Prerequisite: Basic working knowledge of major US sports such as Baseball, Football, Soccer, and Basketball is assumed.

CS 152-SBI Statistical Bioinformatics

Students will learn to estimate probabilities of events through simulation; perform exploratory probabilistic data analysis to identify outliers, artifacts, and patterns; formulate and assess hypotheses about heterogeneous high dimensional data; appropriately visualize data to assess quality and support interpretation; present statistical data analyses clearly and contextually to specific audiences. These aims will be achieved through homework assignments, in class exercises and discussions, readings, and a course project. There will be weekly assignments throughout the term, alternating between problem sets in R and structured parts of the project.

CS 160 Algorithms

Introduction to the study of algorithms. Strategies such as divide-and-conquer, greedy methods, and dynamic programming. Graph algorithms, sorting, searching, integer arithmetic, hashing, and NP-complete problems.

Prerequisite: COMP 15 and COMP/MATH 22 or 61.

CS 163 Computational Geometry

(Cross-listed as MATH 181.)Design and analysis of algorithms for geometric problems. Topics include proof of lower bounds, convex hulls, searching and point location, plane sweep and arrangements of lines, Voronoi diagrams, intersection problems, decomposition and partitioning, farthest-pairs and closest-pairs, rectilinear computational geometry.

Prerequisite: COMP 160, COMP 170, any 100+ MATH course, or permission of the instructor.

CS 166 Computational Systems Biology

Computational modeling of complex biological systems to analyze their emergent properties and understand or predict system behavior. Molecular and protein modeling in the context of biochemical networks. Application of these models across a select set of applications such as metabolomics, gut microbiome analysis, modularity analysis, flux balance analysis, and biological discovery. Introductions to machine learning, linear systems, and statistical concepts that are commonly used in systems biology.

CS 167 Computational Biology

Computational challenges in molecular biology, including sequence alignment and comparison, genomic annotation, micro array data analysis, and proteomics. Underlying computational techniques such as dynamic programming, hidden Markov models, statistical analyses, and search and optimization procedures. Prerequisites: Comp15 and at least one CS course numbered 100 or higher.

Prerequisite: Comp15 and at least one CS course numbered 100 or higher.

CS 168 Convex Optimization

(Cross-listed with EE 109) Convex optimization theory and algorithms. Convex sets, convex functions and convex optimization problems; duality theory and optimality conditions; algorithms for solving convex problems including descent, gradient descent, Newton and interior point methods. Examples of application taken from communications, signal processing and other fields. Project.

Prerequisite: Math 70 or graduate standing.

CS 169 Statistical Bioinformatics

Computational methods and analyses in the context of bioinformatics and biomedical data. Statistical methods in bioinformatics and biomedicine, including the heterogeneous high-dimensional data that tends to arise in this context. Assessment of hypotheses and data quality, visualization, and identification of patterns, outliers, and artifacts of bioinformatics data. Probability estimation through simulation. Presentation of statistical results to specific audiences.

Prerequisite: Recommendations: CS 11 and (CS/Math 61 or Math 65), or graduate standing.

CS 170 Computation Theory

Models of computation: Turing machines, pushdown automata, and finite automata. Grammars and formal languages including context-free languages and regular sets. Important problems including the halting problem and language equivalence theorems.

Prerequisite: COMP 15 and COMP/MATH 22 or 61.

CS 171 Human-Computer Interaction

Introduction to human-computer interaction, or how computers communicate with people. Methodology for designing and testing user interfaces, interaction styles (command line, menus, graphical user interfaces, virtual reality), interaction techniques (including use of voice, gesture, eye movement), design guidelines, and user interface management software system. Students will design a small user interface, program a prototype, and test the result for usability.

Prerequisite: COMP 15

CS 175 Computer Graphics

This course explores the fundamentals of computer graphics, including representing digital images, 2D rasterization and anti-aliasing, 3D rendering via ray casting, ray tracing and radiosity, viewing transformations, 3D shape representation, and an introduction to modeling and computer animation. Assignments and projects require a good working knowledge of the C programming language.

Prerequisite: Recommendations: MATH 42 or 44, and MATH 70 or 72, or graduate standing.

CS 178 Visual Analytics

Visual analytics is the science of combining interactive visual interfaces with automatic data science, machine learning, and AI algorithms to support analytical reasoning. Modern visual analytics tools help users synthesize information and derive insight from large, dynamic, ambiguous, and often conflicting data, and to communicate their findings effectively for decision-making. This course will serve as an introduction to the topic of visual analytics that will include lectures on both theoretical foundations and application methodologies. The goals of this course are for students to: (1) learn about using visual analytics tools (e.g. Tableau), (2) become proficient in generating visualizations within popular data science tools

(e.g. R, Python and scikit-learn), (3) develop their own visual analytics tools (in Javascript and D3), and (4) design evaluation methods to assess the effectiveness of these tools.

CS 182 Cyber in the Civilian Sector

There is a myth that the Internet erases borders. But as Internet companies' ability to place localized ads show, that's false. What's more accurate is that the Internet complicates a nation's ability to control of the flow of information within its borders. (This is not a new challenge for sovereign nations; consider the telegraph.) This fluidity has created great economic opportunity and simplified trans-border access, the latter potentially threatening security and other basic state functions. With bits increasingly controlling the world around us, the Digital Revolution poses a highly disruptive threat. In this course, well explore cyber clashes in the civilian sector: from jurisdictional issues and the challenges posed by new technologies to criminal activities and impacts on civil infrastructures. While several of the topics are also covered in International Cyber Conflict: An Introduction to Power and Conflict in Cyberspace, DHP P249, the intersection between the two courses will be relatively minimal. Cyber in the Civilian Sector will have a greater focus on technology and, naturally enough, on the civilian, as opposed to national-security, side of the house.

CS 183 Privacy in the Digital Age

This course will provide an introduction to the legal and regulatory protections for personal data and the evolving nature of digital surveillance and online privacy. The class will cover public and private sector threats to privacy and look at how different countries have implemented both privacy protections and surveillance regimes that affect individual Internet users and govern the collection of their personal data. Topics to be covered include encryption policy, law enforcement access to data, intelligence agency access to data, the European General Data Protection Regulation, the Electronic Communications Privacy Act, domestic and international lawful interception of data, protections for geolocation data and other forms of metadata, and facial recognition technology. This course will primarily focus on law and policy measures related to privacy but it will also cover some basic technical material related to cryptography and networking that is relevant to understanding the impacts of different policies.

CS 184 Cyberlaw and Cyberpolicy

This course is an introduction to the legal issues of cyberspace. Legal issues in this domain are complex. Technology has been evolving faster than the law's ability to handle the changes, so partially this course will be an education in the legislating and policy making of moving targets. It will also be an education in jurisdiction, privacy, surveillance, and copyright as it relates to the Internet. The perspective will be from US law and jurisprudence, although there will be periodic forays into international issues. Topics covered will include cyberlaw and cybergovernance, the Digital Revolution and its impact on First, Fourth, and Fifth Amendment issues, copyright in the Digital Age, and the Computer Fraud and Abuse Act.

CS 185 Computing for Developing Regions

An interdisciplinary approach to the role of computing technologies in developing regions. Low-cost communication infrastructure; socially relevant technologies for education, healthcare, and governance; and the use of technology by underserved communities and populations in developing regions. Problems and existing solutions covered through case studies. Group projects on designing, implementing, and evaluating a solution. Recommendations: CS10 or CS11 or some background in computer science and/or technology.

Prerequisite: Completion of COMP 15 or graduate standing

CS 191 Research

Research on a topic in Computer Science or a related discipline, culminating in a final paper describing accomplishments, with the goal of advancing the state of the art. Topic is proposed by a faculty sponsor in Computer Science. Faculty consent required. Students sign up for a section that corresponds to a faculty member.

Prerequisite: Consent

CS 193 Directed Study

Guided study of an approved topic. Credit as arranged.

Prerequisite: Consent.

CS 193-CC Computational Complexity

TBD

Prerequisite: Consent

CS 193-GFA Geometric Folding Algorithms

TBD

Prerequisite: Consent of department

CS 193-MSC MS CoreComp

TBD

Prerequisite: Consent of department

CS 193-ROS Introduction to ROS

CS 197 Honors Thesis

Honors Thesis Computer Science.

DS 93 Directed Study

Directed study, as approved by professor and department.

DS 97 Senior Capstone Project in Data Science I

Application of data science and analytic principles to the solution of a real-world problem in a group setting. Requirements analysis, review of available data sources, and proposal of a solution strategy to the problem.

Prerequisite: Senior standing

DS 98 Senior Capstone Project in Data Science II

A continuation of COMP 87. Analysis of the problem proposed in COMP 87 is completed and a final paper summarizes data gathered, analytic results, lessons learned, and opportunities for future study.

Prerequisite: COMP 87

DS 143 Data Science for Sustainability

Crosslisted as ME 193. This course explores emerging topics in data science and statistical learning with applications to the three pillars of sustainability (environmental, economic, and social). Students learn to build, estimate, and interpret models that describe phenomena in the broad area of energy and environmental decision-making with an emphasis on social justice. Students leave the course as both critical consumers and responsible producers of data-driven analysis. The objectives of this class include i) learning a suite of data-driven modeling and prediction tools, ii) building the programming and computing expertise to use those tools, and iii) developing the ability to formulate an analysis to answer sustainability questions of interest to industry and/or government partners.

Prerequisite: This course uses Python. Prior experience with statistics and programming is required.

DS 153-AEP Artificial Intelligence: Algorithms, Ethics, and Policy

Artificial intelligence (AI) has emerged as a transformative technology that is being adopted in a wide range of settings with significant impact on society. This course will examine the ethics of AI usage in these settings and how policy can

affect this usage. The course will also introduce technical aspects of AI algorithms. With combined perspectives on policy and technical knowledge, students who complete the course will be equipped to become informed policy experts and decision-makers with substantial understanding of the technology necessary to lead on issues involving AI. This new course is still under development, so details are TBA and subject to change.

Note: Cross listed with CS 153-01.

DS 153-AISI Generative AI for Social Impact

Prerequisite: Completion of CS 15 or graduate standing.

DS 153-CVI Computer Vision

This course is an introduction to low and intermediate level Computer Vision. We will learn how to design algorithms that process visual scenes to automatically extract information. The course will cover fundamental principles and important applications of computer vision, including image formation, processing, detection and matching features, image segmentation, and multiple views.

DS 153-MAS Multi-Agent Systems

Prerequisite: Prior completion of CS 131

DS 153-PHD Probability in High Dimension

TBA

DS 153-SUR Surveillance

What does it mean to do data science in a society filled with competing economic, social, and political interests? In this course, we will draw from historical perspectives on present-day issues in computer science to develop a social and political understanding of what the problem is, how it came to be, and what we can do about it. Our topic of focus will be surveillance, a problem that intersects economics through capitalism, race through histories of the policing of Black and Brown people, and power through systems of control. Throughout the three arcs comprising the class – (1) defining surveillance and learning its history, (2) developing an ethical inquiry and critique of surveillance technology, and (3) applying this practice to present day related problems – we will build “muscles” of analysis and practice that will build a foundation for budding data scientists to navigate a social and political world.

DS 153-THR Trust in Human-Robot Interaction

This class takes a cognitive science-focused look at how humans trust robots. We'll start by defining trust as it relates to robots (and compared to humans or other technology), and throughout the semester cover topics such as: what factors affect different types of trust, how do we measure trust both subjectively and behaviorally, what may happen if a robot is over trusted or under trusted, how can we model trust computationally, and what ethical issues can be tangled up with trying to build a trustworthy system. This will be a primarily discussion-based class, and the main class project will be to come up with a research project about trust in human-robot interaction and implement it in a human-robot interaction system.

Prerequisite: Completion of CS 133 recommended

DS 154-NLP Natural Language Processing & Human Records

TBA

DS 154-SBI Statistical Bioinformatics

Students will learn to estimate probabilities of events through simulation; perform exploratory probabilistic data analysis to identify outliers, artifacts, and patterns; formulate and assess hypotheses about heterogeneous high dimensional data; appropriately visualize data to assess quality and support interpretation; present statistical data analyses clearly and contextually to specific audiences. These aims will be achieved through homework assignments, in class exercises and discussions, readings, and a course project. There will be weekly assignments throughout the term, alternating between problem sets in R and structured parts of the project.

DS 193 Directed Study

Directed study, as approved by professor and department.

ES 2-ICE Introduction to Computing in Engineering

An introduction to engineering problem-solving with the aid of computational software. Scientific computing concepts will be introduced including number representation, arrays, structured programming techniques, and good coding practices. Basic numerical and data analysis methods will be introduced including numerical differentiation and integration, matrix operations, descriptive statistics, curve fitting, and optimization. Examples drawn from a variety of engineering disciplines will give students extensive practice in coding solutions and applying them to data.

Prerequisite: MATH 32 recommended

Back to Main Courses Page