COMPUTER SCIENCE (Div III)

Chair: Professor Stephen Freund

- Jeannie R Albrecht, Professor of Computer Science; on leave 2020-2021
- Duane A. Bailey, Professor of Computer Science
- Daniel W. Barowy, Assistant Professor of Computer Science; on leave 2020-2021
- Rohit Bhattacharya, Assistant Professor of Computer Science
- Andrea Danyluk, Mary A and William Wirt Warren Professor of Computer Science, Chair of Cognitive Science Program; affiliated with: Cognitive Science Program
- Molly Q Feldman, Visiting Assistant Professor of Computer Science
- Stephen N. Freund, Chair & John B. McCoy and John T. McCoy Professor of Computer Science
- Iris Howley, Assistant Professor of Computer Science
- Bill K. Jannen, Assistant Professor of Computer Science
- William J. Lenhart, A. Barton Hepburn Professor of Computer Science
- Samuel McCauley, Assistant Professor of Computer Science
- Anna C. Neufeld, Visiting Lecturer in Computer Science
- Kelly A. Shaw, Associate Professor of Computer Science
- Shikha Singh, Assistant Professor of Computer Science
- Aaron M. Williams, Assistant Professor of Computer Science

Computers and computation are pervasive in our society. They play enormously important roles in areas as diverse as education, science, business, and the arts. Understanding the nature of computation and exploring the great potential of computers are the goals of the discipline of computer science. A sample of the areas of research investigated by the Williams Department of Computer Science alone illustrates the vast range of topics that are of interest to computer scientists and computing professionals today. This includes: the use of computer-generated graphic images in the arts and as a tool for visualization in the sciences and other areas; the protocols that make transmission of information over the Internet possible; the design of revolutionary new computer languages that simplify the process of constructing complex programs for computers; the development of machine learning algorithms that can extract useful and even novel information from data that is too complex for humans to analyze; algorithms that can solve problems that were previously too hard to solve in a reasonable amount of time, just by giving up a little bit of optimality in the solution; the investigation of machine architectures and specific hardware aimed at making computing fast.

The department recognizes that students' interests in computer science will vary widely. The department attempts to meet these varying interests through: (1) the major; (2) a selection of courses intended for those who are interested primarily in an introduction to computer science; (3) recommended course sequences for the non-major who wants a more extensive introduction to computer science in general or who seeks to develop some specific expertise in computing for application in some other discipline.

MAJOR

The goal of the major is to provide an understanding of algorithmic problem solving as well as the conceptual organization of computers and complex programs running on them. Emphasis is placed on the fundamental principles of computer science, building upon the mathematical and theoretical ideas underlying these principles. The introductory and core courses build a broad and solid base for understanding computer science. The more advanced courses allow students to sample a variety of specialized areas including graphics, artificial intelligence, computer architecture, networks, compiler design, human computer interaction, distributed systems, and operating systems. Independent study and honors work provide opportunities for students to study and conduct research on topics of special interest.

The major in Computer Science equips students to pursue a wide variety of career opportunities. It can be used as preparation for a career in computing, for graduate school, or to provide important background and techniques for the student whose future career will extend outside of computer science.

MAJOR REQUIREMENTS

Required Courses in Computer Science

A minimum of 8 courses is required in Computer Science, including the following:

Introductory Courses

Computer Science 134 Introduction to Computer Science

Computer Science 136 Data Structures and Advanced Programming

Core Courses

Computer Science 237 Computer Organization

Computer Science 256 Algorithm Design and Analysis

Computer Science 334 Principles of Programming Languages

Computer Science 361 Theory of Computation

Elective Courses

Two or more electives (bringing the total number of Computer Science courses to at least 8) chosen from 300- or 400-level courses in Computer Science. Computer Science courses with 9 as the middle digit (reading, research, and thesis courses) will normally not be used to satisfy the elective requirements. Students may petition the department to waive this restriction with good reason.

Required Courses in Mathematics

Any Mathematics or Statistics course at the 200-level or higher except for MATH 200

Required Proficiency in Discrete Mathematics

Students must demonstrate proficiency in discrete mathematics by either passing the departmental Discrete Mathematics Proficiency Exam or by earning a grade of C- or better in MATH 200. This requirement must be met by the end of the sophomore year.

The Discrete Mathematics Proficiency Exam may be taken at most twice and cannot be taken beyond the sophomore year. The exam may not be used to fulfill the requirement for a student who has taken the course pass/fail or who has received a letter grade below C- in Math 200.

Students considering pursuing a major in Computer Science are urged to take Computer Science 134 and to begin satisfying their mathematics requirements early. Note in particular that the Discrete Mathematics Proficiency requirement is a prerequisite for many advanced courses.

Students who take Computer Science 102T, 103, 107, or 109 may use that course as one of the two electives required for the major in Computer Science. Computer Science 102T, 103, 107, 109, and 134 are not open to students who have taken a Computer Science course numbered 136 or higher.

To be eligible for admission to the major, a student must have completed at least two Computer Science courses, including Computer Science 136, as well as fulfilled the Discrete Mathematics Proficiency Requirement by the end of the sophomore year. A Mathematics course at the 200-level or higher (except for MATH 200) must be completed by the end of the junior year. Students are urged to have completed two of the four core courses (Computer Science 237, 256, 334, and 361) by the end of the sophomore year and must normally have completed at least three out of the four core courses by the end of the junior year.

We encourage students to be intellectually engaged in our field beyond the formal structure of courses. As such, all computer science majors must attend at least twenty Computer Science colloquia. Juniors and seniors are encouraged to attend at least five during each semester they are present on campus. Prospective majors in their first and second years are also encouraged to attend.   A student studying away on a program approved by the International Education and Study Away Office will receive four colloquium credits for each semester away, up to a total of eight credits.

With the advance permission of the department, two appropriate mathematics or statistics courses may be substituted for one Computer Science elective. Appropriate mathematics classes are those numbered 300 or above, and appropriate statistics courses are those numbered 200 or above. Other variations in the required courses, adapting the requirements to the special needs and interests of the individual student, may be arranged in consultation with the department.

LABORATORY FACILITIES

The Computer Science Department maintains five departmental computer laboratories for students taking Computer Science courses, as well as a lab that can be configured for teaching specialized topics such as robotics. The workstations in these laboratories also support student and faculty research in computer science.

THE DEGREE WITH HONORS IN COMPUTER SCIENCE

The degree with honors in Computer Science is awarded to students who have demonstrated outstanding intellectual achievement in a program of study extending beyond the requirements of the regular major. The principal considerations in recommending a student for the degree with honors will be: mastery of core material, ability to pursue independent study of computer science, originality in methods of investigation, and creativity in research.

Honors study is highly recommended for those students with strong academic records in computer science who wish to attend graduate school, pursue high-level industrial positions in computing, or who would simply like to experience research in computer science.

Prospective honors students are urged to consult with their departmental advisor at the time of registration in the spring of the sophomore or at the beginning of the junior year to arrange a program of study that could lead to the degree with honors. Such a program normally consists of Computer Science 493 and 494 and a WSP of independent research under the guidance of a Computer Science faculty member, culminating in a thesis that is judged acceptable by the department. The program produces a significant piece of written work and often includes a major computer program. All honors candidates are required to give an oral presentation of their research in the Computer Science Colloquium in early spring semester.

Students considering honors work should obtain permission from the department before registering in the fall of the senior year. Formal admission to candidacy occurs at the beginning of the spring semester of the senior year and is based on promising performance in the fall semester and winter study units of honors work. Recommendations for the degree with honors will be made for outstanding performance in the three honors courses. Highest honors will be recommended for students who have displayed exceptional ability, achievement, or originality.

INTRODUCTORY COURSES

The department offers a choice of five introductory courses; Computer Science 102 The Socio-Techno Web, 103 Electronic Textiles, 107 Creating Games, 109 The Art and Science of Computer Graphics, and 134 Introduction to Computer Science.

Computer Science 134 provides an introduction to computer science with a focus on developing computer programming skills. These skills are essential to most upper-level courses in the department. As a result, Computer Science 134 together with Computer Science 136, are required as a prerequisite to most advanced courses in the department. Those students intending to take several Computer Science courses are urged to take 134 early.

Those students interested in learning more about exciting new ideas in computer science, but not necessarily interested in developing extensive programming skills, should consider Computer Science 102 The Socio-Techno Web, 103 Electronic Textiles, 107 Creating Games, or 109 The Art and Science of Computer Graphics.

Students with significant programming experience should consider electing Computer Science 136 (see “Advanced Placement” below). Students are always welcome to contact a member of the department for guidance in selecting a first course.

COMPUTER SCIENCE 134

Introduction to Computer Science covers fundamental concepts in the design, implementation and testing of computer programs including loops, conditionals, functions, elementary data types and recursion. There is a strong focus on constructing correct, understandable and efficient programs in a structured language such as Java or Python.

STUDY ABROAD

Study abroad can be a wonderful experience. Students who hope to take computer science courses while abroad should discuss their plans in advance with the chair of the department or the departmental study away advisor. Students who plan to study away but do not expect to take courses toward the major should work with the department to create a plan to ensure that they will be able to complete the major. While study abroad is generally not an impediment to completing the major, students should be aware that certain computer science courses must be taken in a particular sequence and that not all courses are offered every semester (or every year). Students who wish to discuss their plans are invited to meet with any of the faculty in Computer Science.

FAQ

Students MUST contact departments/programs BEFORE assuming study away credit will be granted toward the major or concentration.

Can your department or program typically pre-approve courses for major/concentration credit?

Yes, in some cases, if appropriate course information is available in advance (e.g. syllabi and/or course descriptions), though students should be sure to contact the department.

What criteria will typically be used/required to determine whether a student may receive major/concentration credit for a course taken while on study away?

Course title and description, and complete syllabus, including readings and assignments.

Does your department/program place restrictions on the number of major/concentration credits that a student might earn through study away?

Yes. Typically no more than two CSCI courses and one Math course.

Does your department/program place restrictions on the types of courses that can be awarded credit towards your major?

No.

Are there specific major requirements that cannot be fulfilled while on study away?

No.

Are there specific major requirements in your department/program that students should be particularly aware of when weighing study away options? (Some examples might include a required course that is always taught in one semester, laboratory requirements.)

Yes. Many CSCI electives are not taught every year. Students should develop a plan to complete all major requirements and discuss them with the department prior to going abroad.

Give examples in which students thought or assumed that courses taken away would count toward the major or concentration and then learned they wouldn't:

Students must have courses pre-approved prior to going abroad to ensure they meet the curricular goals and standards of the department.

ADVANCED PLACEMENT

Students with an extensive background in computer science are urged to take the Advanced Placement Examination in Computer Science. A score of 4 or better on the AP Computer Science A exam is normally required for advanced placement in Computer Science 136.

Students who wish to be placed in Computer Science 136 but who have not taken the Advanced Placement Examination should consult with the department. Such students should have had a good course in computer science using a structured language such as Java or Python.

PLANS OF STUDY FOR NON-MAJORS

The faculty in Computer Science believes that students can substantially enrich their academic experience by completing a coherent plan of study in one or more disciplines outside of their majors. With this in mind, we have attempted to provide students majoring in other departments with options in our department's curriculum ranging from two-course sequences to collections of courses equivalent to what would constitute a minor at institutions that recognize such a concentration. Students interested in designing such a plan of study are invited to discuss their plans in detail with a member of the faculty. To assist students making such plans, we include some suggestions below.

Students seeking to develop an extensive knowledge of computer science without majoring in the department are encouraged to use the major requirements as a guide. In particular, the four core courses required of majors are intended to provide a broad knowledge of topics underlying all of computer science. Students seeking a concentration in Computer Science are urged to complete at least two of these courses followed by one of our upper-level electives. Such a program would typically require the completion of a total of five Computer Science courses in addition to the Discrete Mathematics Proficiency requirement.

There are several sequences of courses appropriate for those primarily interested in developing skills in programming for use in other areas. For general programming, Computer Science 134 followed by 136 and 256 will provide students with a strong background in algorithm and data structure design together with an understanding of issues of correctness and efficiency. Students of the Bioinformatics program are encouraged to take Computer Science 134 at a minimum, and should also consider Computer Science 136 and 256. The sequence of courses Computer Science 109 and 134 would provide sufficient competence in computer graphics for many students interested in applying such knowledge either in the arts or sciences.

There are, of course, many other alternatives. We encourage interested students to consult with the department chair or other members of the department's faculty.

GENERAL REMARKS

Divisional Requirements

All Computer Science courses may be used to satisfy the Division III distribution requirement.

Alternate Year Courses

Computer Science 102T, 103, 107, 109, 315, 319, 326, 331, 333, 336T, 337T, 338, 339, 356T, 357, 358, 371, 373, 374T, 375, 376, 432, and 434T are each usually offered every other year. All other Computer Science courses are normally offered every year.

Course Numbering

The increase from 100, through 200 and 300, to 400 indicates in most instances an increasing level of maturity in the subject that is expected of students. Within a series, numeric order does not indicate the relative level of difficulty of courses. Rather, the middle digit of the course number (particularly in upper-level courses) generally indicates the area of computer science covered by the course.

Course Descriptions

Brief descriptions of the courses in Computer Science can be found below. More detailed information on the offerings in the department is available at http://www.cs.williams.edu/.

Courses Open on a Pass-Fail Basis

Students taking a Computer Science course on a pass-fail basis must meet all the requirements set for students taking the course on a graded basis.

With the permission of the department, any course offered by the department may be taken pass-fail (with the exception of tutorials), though courses graded with the pass-fail option may not be used to satisfy any of the major or honors requirements. However, with the permission of the department, courses taken in the department beyond those requirements may be taken on a pass-fail basis.

CSCI 102 (F) The Socio-Techno Web (QFR)

This course introduces many fundamental concepts in computer science by examining the social aspects of computing. As more and more people use the technologies and services available via the Internet, online environments like Facebook, Amazon, Google, Twitter, and blogs are flourishing. However, several of the problems related to security, privacy, and trust that exist in the real world transfer and become amplified in the virtual world created by the ubiquity and pervasiveness of the Internet. In this course, we will investigate how the social, technological, and natural worlds are connected, and how the study of networks sheds light on these connections. Topics include the structure of the Social Web and networks in general; issues such as virtual identity, personal and group privacy, trust evaluation and propagation, and online security; and the technology, economics, and politics of Web information and online communities. No background in computer science or programming is required or expected.

Class Format: groups of three or four

Requirements/Evaluation: tutorial discussions, presentations, problem sets and labs, a midterm exam, and a final project or paper

Prerequisites: none

Enrollment Limit: 18

Enrollment Preferences: first-year students and sophomores who have not previously taken a computer science course

Expected Class Size: 18

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 103 (F) Electronic Textiles (QFR)

Digital data is being infused throughout the entire physical world, escaping the computer monitor and spreading to other devices and appliances, including the human body. Electronic textiles, or eTextiles, is one of the next steps toward making everything interactive and this course aims to introduce learners to the first steps of developing their own wearable interactive technology devices. After completing a series of introductory eTextiles projects to gain practice in necessary sewing, circuitry, and programming skills, students will propose and design their own eTextiles projects, eventually implementing them with sewable Arduino components, and other found electronic components as needed. The scope of the project will depend on the individual's prior background, but can include everything from a sweatshirt with light-up turn signals for bicycling, to a wall banner that displays the current air quality of the room, to a stuffed animal that plays a tune when the lights go on, to whatever project you can conceivably accomplish with sewable Arduino inputs, outputs, and development board in a semester context. This class will introduce students to introductory computer programming, circuitry, and sewing with the goal of creating novel wearable artifacts that interact with the world.

Class Format: interspersed with hands-on activities in a computer lab

Requirements/Evaluation: weekly homework assignments and a final project

Prerequisites: none

Enrollment Limit: 20

Enrollment Preferences: students who have not previously taken a CSCI course

Expected Class Size: 20

Grading: yes pass/fail option, yes fifth course option

Materials/Lab Fee: a fee of $85 will be added to term bill to cover Lilypad Arduino components (Protosnap Plus Kit, battery holders switched and not-switched, sets of LEDs, temperature sensor, vibe board, tri-color LED), alligator test leads, and fabric scissors

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will teach students the basics of computer programming through projects in which quantitative/formal reasoning skills are practiced and evaluated.

Not offered current academic year

CSCI 134 (F)(S) Introduction to Computer Science (QFR)

This course introduces students to the science of computation by exploring the representation and manipulation of data and algorithms. We organize and transform information in order to solve problems using algorithms written in a modern object-oriented language. Topics include organization of data using objects and classes, and the description of processes using conditional control, iteration, methods and classes. We also begin the study of abstraction, self-reference, reuse, and performance analysis. While the choice of programming language and application area will vary in different offerings, the skills students develop will transfer equally well to more advanced study in many areas. In particular, this course is designed to provide the programming skills needed for further study in computer science and is expected to satisfy introductory programming requirements in other departments.

Requirements/Evaluation: weekly programming projects, weekly written homeworks, and two examinations.

Prerequisites: none, except for the standard prerequisites for a (QFR) course; previous programming experience is not required

Enrollment Limit: 30(10/lab)

Enrollment Preferences: if the course is over-enrolled, enrollment will be determined by lottery

Expected Class Size: 30

Grading: yes pass/fail option, yes fifth course option

Unit Notes: students with prior experience with object-oriented programming should discuss appropriate course placement with members of the department

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This course include regular and substantial problem sets, labs, and/or projects in which quantitative/formal reasoning skills are practiced and evaluated.

Attributes: BIGP Courses COGS Interdepartmental Electives

Fall 2020

LEC Section: 02 MWF 12:00 pm - 12:50 pm Daniel P. Aalberts

LEC Section: R1 MWF 9:20 am - 10:10 am Duane A. Bailey

LAB Section: R3 MR 1:30 pm - 2:45 pm Duane A. Bailey

LAB Section: R4 MR 1:30 pm - 2:45 pm Molly Q Feldman

LAB Section: R5 MR 3:15 pm - 4:30 pm Duane A. Bailey

LAB Section: R6 MR 3:15 pm - 4:30 pm Molly Q Feldman

LAB Section: R7 TR 9:45 am - 11:00 am Daniel P. Aalberts

LAB Section: R8 TF 3:15 pm - 4:30 pm Daniel P. Aalberts

Spring 2021

LEC Section: R1 MWF 9:20 am - 10:10 am Duane A. Bailey, Molly Q Feldman

LEC Section: R2 MWF 10:40 am - 11:30 am Duane A. Bailey, Molly Q Feldman

LAB Section: R3 M 1:30 pm - 2:45 pm Molly Q Feldman

LAB Section: R4 M 1:30 pm - 2:45 pm Duane A. Bailey

LAB Section: R5 M 3:15 pm - 4:30 pm Duane A. Bailey

LAB Section: R6 M 3:15 pm - 4:30 pm Duane A. Bailey

LAB Section: R7 T 9:45 am - 11:00 am Duane A. Bailey

LAB Section: R8 T 3:15 pm - 4:30 pm Duane A. Bailey

CSCI 136 (F)(S) Data Structures and Advanced Programming (QFR)

This course builds on the programming skills acquired in Computer Science 134. It couples work on program design, analysis, and verification with an introduction to the study of data structures. Data structures capture common ways in which to store and manipulate data, and they are important in the construction of sophisticated computer programs. Students are introduced to some of the most important and frequently used data structures: lists, stacks, queues, trees, hash tables, graphs, and files. Students will be expected to write several programs, ranging from very short programs to more elaborate systems. Emphasis will be placed on the development of clear, modular programs that are easy to read, debug, verify, analyze, and modify.

Class Format: Lecture content will be through asynchronously viewed video modules. Three scheduled (MWF) course sections will be used for synchronous conference meetings. Two sections will be in-person and one will be remote. There will be 5 scheduled weekly lab sections that will be remote. Students should sign up for the lecture section, one conference, and one lab.

Requirements/Evaluation: programming and written assignments, quizzes, examinations

Prerequisites: CSCI 134 or equivalent; fulfilling the Discrete Mathematics Proficiency requirement is recommended, but not required

Enrollment Limit: 60(12/lab)

Enrollment Preferences: if the course is over-enrolled, enrollment will be determined by lottery

Expected Class Size: 60

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This course include regular and substantial problem sets, labs, and/or projects in which quantitative/formal reasoning skills are practiced and evaluated.

Attributes: BIGP Courses

Fall 2020

CON Section: 03 MWF 10:40 am - 11:30 am Bill K. Jannen

LEC Section: R1 TBA Bill K. Jannen, William J. Lenhart

CON Section: R2 MWF 9:20 am - 10:10 am William J. Lenhart

CON Section: R4 MWF 12:00 pm - 12:50 pm William J. Lenhart

LAB Section: R5 R 1:00 pm - 2:30 pm William J. Lenhart

LAB Section: R6 R 1:00 pm - 2:30 pm Bill K. Jannen

LAB Section: R7 R 3:30 pm - 5:00 pm William J. Lenhart

LAB Section: R8 R 3:30 pm - 5:00 pm Bill K. Jannen

LAB Section: R9 R 8:30 pm - 10:00 pm Bill K. Jannen, William J. Lenhart

Spring 2021

CON Section: H3 MWF 10:40 am - 11:30 am Samuel McCauley

CON Section: H4 MWF 12:00 pm - 12:50 pm Samuel McCauley

LEC Section: R1 ASYN William J. Lenhart, Samuel McCauley

CON Section: R2 MWF 9:20 am - 10:10 am William J. Lenhart

LAB Section: R5 R 1:00 pm - 2:30 pm Samuel McCauley

LAB Section: R6 R 1:00 pm - 2:30 pm William J. Lenhart

LAB Section: R7 R 3:30 pm - 5:00 pm Samuel McCauley

LAB Section: R8 R 3:30 pm - 5:00 pm William J. Lenhart

LAB Section: R9 R 9:45 am - 11:15 am William J. Lenhart, Samuel McCauley

CSCI 237 (F)(S) Computer Organization (QFR)

This course studies the basic instruction set architecture and organization of a modern computer. It provides a programmer's view of how computer systems execute programs, store information, and communicate. Over the semester the student learns the fundamentals of translating higher level languages into assembly language, and the interpretation of machine languages by hardware. At the same time, a model of computer hardware organization is developed from the gate level upward.

Class Format: There is no scheduled time for lectures. They will be available for asynchronous viewing. Each lab section will meet once per week. Students should sign up for lecture and one lab.

Requirements/Evaluation: weekly programming assignments and/or problem sets, midterm and final exams

Prerequisites: CSCI 136

Enrollment Limit: 24(8/lab)

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 20

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will consist of programming assignments and problem sets in which quantitative/formal reasoning skills are practiced and evaluated.

Fall 2020

LAB Section: 02 TF 1:30 pm - 2:45 pm Kelly A. Shaw

LAB Section: 03 TF 3:15 pm - 4:30 pm Kelly A. Shaw

LAB Section: 04 WF 1:30 pm - 2:45 pm Kelly A. Shaw

LAB Section: 05 MW 10:00 am - 11:15 am Kelly A. Shaw

LAB Section: 06 MWF 11:45 am - 1:00 pm Kelly A. Shaw

LEC Section: R1 TBA Kelly A. Shaw

LAB Section: R7 MWF 8:15 am - 9:30 am Kelly A. Shaw

Spring 2021

LAB Section: 03 W 11:45 am - 1:00 pm Kelly A. Shaw

LAB Section: 04 W 1:30 pm - 2:45 pm Kelly A. Shaw

LEC Section: R1 TBA Kelly A. Shaw

LAB Section: R2 W 8:15 am - 9:30 am Kelly A. Shaw

CSCI 256 (F)(S) Algorithm Design and Analysis (QFR)

This course investigates methods for designing efficient and reliable algorithms. By carefully analyzing the structure of a problem within a mathematical framework, it is often possible to dramatically decrease the computational resources needed to find a solution. In addition, analysis provides a method for verifying the correctness of an algorithm and accurately estimating its running time and space requirements. We will study several algorithm design strategies that build on data structures and programming techniques introduced in Computer Science 136. These include induction, divide-and-conquer, dynamic programming, and greedy algorithms. Additional topics of study include algorithms on graphs and strategies for handling potentially intractable problems.

Class Format: Lectures will be simultaneously recorded in classroom and broadcast over Zoom. Office hours will be done over Zoom. Some additional course materials (examples, solutions, definitions and core concepts, etc.) may be provided as prerecorded videos.

Requirements/Evaluation: Problem sets, midterm and final examinations

Prerequisites: CSCI 136 and fulfillment of the Discrete Mathematics Proficiency requirement

Enrollment Limit: 20

Enrollment Preferences: Preference will be given to students who need the class in order to complete the major. Ties will be broken by seniority (seniors first, then juniors, etc.).

Expected Class Size: 20

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This course will have weekly problem sets in which students will formally prove statements about the behavior and performance of algorithms. In short, the entirety of the course is about applying abstract and mathematical reasoning to the way computers work.

Fall 2020

LEC Section: H1 MWF 10:40 am - 11:30 am Samuel McCauley

Spring 2021

LEC Section: H1 MWF 10:40 am - 11:30 am Shikha Singh

LEC Section: H2 MWF 12:00 pm - 12:50 pm Shikha Singh

CSCI 315 (S) Computational Biology (QFR)

Cross-listings: PHYS 315 CSCI 315

Secondary Cross-listing

This course will provide an overview of Computational Biology, the application of computational, mathematical, statistical, and physical problem-solving techniques to interpret the rapidly expanding amount of biological data. Topics covered will include database searching, DNA sequence alignment, clustering, RNA structure prediction, protein structural alignment, methods of analyzing gene expression, networks, and genome assembly using techniques such as string matching, dynamic programming, hidden Markov models, and expectation-maximization.

Requirements/Evaluation: weekly Python programming assignments, problem sets, a few quizzes and a final project

Prerequisites: programming experience (e.g., CSCI 136), mathematics (PHYS/MATH 210 or MATH 150), and physical science (PHYS 142 or 151, or CHEM 151 or 153 or 155), or permission of instructor

Enrollment Limit: 10

Enrollment Preferences: based on seniority

Expected Class Size: 8

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

This course is cross-listed and the prefixes carry the following divisional credit:

PHYS 315 (D3) CSCI 315 (D3)

Attributes: BIGP Courses

Not offered current academic year

CSCI 319 (S) Integrative Bioinformatics, Genomics, and Proteomics Lab (QFR)

Cross-listings: MATH 319 CHEM 319 BIOL 319 PHYS 319 CSCI 319

Secondary Cross-listing

What can computational biology teach us about cancer? In this lab-intensive experience for the Genomics, Proteomics, and Bioinformatics program, computational analysis and wet-lab investigations will inform each other, as students majoring in biology, chemistry, computer science, mathematics/statistics, and physics contribute their own expertise to explore how ever-growing gene and protein data-sets can provide key insights into human disease. In this course, we will take advantage of one well-studied system, the highly conserved Ras-related family of proteins, which play a central role in numerous fundamental processes within the cell. The course will integrate bioinformatics and molecular biology, using database searching, alignments and pattern matching, and phylogenetics to reconstruct the evolution of gene families by focusing on the gene duplication events and gene rearrangements that have occurred over the course of eukaryotic speciation. By utilizing high through-put approaches to investigate genes involved in the inflammatory and MAPK signal transduction pathways in human colon cancer cell lines, students will uncover regulatory mechanisms that are aberrantly altered by siRNA knockdown of putative regulatory components. This functional genomic strategy will be coupled with independent projects using phosphorylation-state specific antisera to test our hypotheses. Proteomic analysis will introduce the students to de novo structural prediction and threading algorithms, as well as data-mining approaches and Bayesian modeling of protein network dynamics in single cells. Flow cytometry and mass spectrometry may also be used to study networks of interacting proteins in colon tumor cells.

Class Format: two afternoons of lab, with one hour of lecture, per week. In most weeks, we will meet one day for lecture discussions.

Requirements/Evaluation: lab participation, several short homework assignments, one lab report, a programming project, and a grant proposal

Prerequisites: BIOL 202; students who have not taken BIOL 202 but have taken BIOL 101 and a CSCI course, or CSCI/PHYS 315, may enroll with permission of instructor. No prior computer programming experience is required.

Enrollment Limit: 12

Enrollment Preferences: seniors, then juniors, then sophomores

Expected Class Size: 12

Grading: yes pass/fail option, yes fifth course option

Unit Notes: does not satisfy the distribution requirement for the Biology major

Distributions: (D3) (QFR)

This course is cross-listed and the prefixes carry the following divisional credit:

MATH 319 (D3) CHEM 319 (D3) BIOL 319 (D3) PHYS 319 (D3) CSCI 319 (D3)

Quantative/Formal Reasoning Notes: Through lab work, homework sets and a major project, students will learn or further develop their skills in programming in Python, and about the basis of Bayesian approaches to phylogenetic tree estimation.

Attributes: BIGP Courses BIMO Interdepartmental Electives

Spring 2021

SEM Section: 01 TR 9:45 am - 11:00 am Lois M. Banta

LAB Section: H3 MW 1:00 pm - 3:00 pm Lois M. Banta

LAB Section: H4 TR 1:00 pm - 3:00 pm Lois M. Banta

LAB Section: H5

SEM Section: R2 MW 6:45 pm - 8:00 pm Lois M. Banta

CSCI 326 (S) Software Methods (QFR)

Sophisticated software systems play a prominent role in many aspects of our lives, and while programming can be a very creative and exciting process, building a reliable software system of any size is no easy feat. Moreover, the ultimate outcome of any programming endeavor is likely to be incomplete, unreliable, and unmaintainable unless principled methods for software construction are followed. This course explores those methods. Specific topics include: software processes; specifying requirements and verifying correctness; abstractions; design principles; software architectures; concurrentand scalable systems design; testing and debugging; and performance evaluation.

Requirements/Evaluation: homework, programming assignments, group work, presentations, exams

Prerequisites: CSCI 136, and at least one of CSCI 237, 256, or 334

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 331 (F) Introduction to Computer Security (QFR)

This class explores common vulnerabilities in computer systems, how attackers exploit them, and how systems engineers design defenses to mitigate them. The goal is to be able to recognize potential vulnerabilities in one's own software and to practice defensive design. Hands-on experience writing C/C++ code to inspect and modify the low-level operation of running programs is emphasized. Finally, regular reading and writing assignments round out the course to help students understand the cultural and historical background of the computer security "arms race."

Requirements/Evaluation: assignments, midterm exam, and final exam

Prerequisites: CSCI 136 and CSCI 237

Enrollment Limit: 24

Enrollment Preferences: upper-level students

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 333 (S) Storage Systems (QFR)

This course will examine topics in the design, implementation, and evaluation of storage systems. Topics include the memory hierarchy; ways that data is organized (both logically and physically); storage hardware and its influence on storage software designs; data structures; performance models; and system measurement/evaluation. Readings will be taken from recent technical literature, and an emphasis will be placed on identifying and evaluating design trade-offs.

Class Format: Lecture content will be through asynchronously viewed video modules. Two scheduled conference sections will each meet twice per week. They will be used for synchronous conference meetings that include discussions, activities, and programming tasks. One conference section will be in-person and one will be remote. Students should sign up for the lecture section and one conference section.

Requirements/Evaluation: programming assignments, quizzes, midterm examination, and a final project

Prerequisites: CSCI 136; CSCI 237 or permission of instructor

Enrollment Limit: 40

Enrollment Preferences: current Computer Science majors, students with research experience or interest

Expected Class Size: 40

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This course will have students develop quantitative/formal reasoning skills through problem sets and programming assignments.

Spring 2021

CON Section: 03 TR 11:30 am - 12:45 pm Bill K. Jannen

LEC Section: R1 ASYN Bill K. Jannen

CON Section: R2 TR 9:45 am - 11:00 am Bill K. Jannen

CSCI 334 (F)(S) Principles of Programming Languages (QFR)

This course examines the concepts and structures governing the design and implementation of programming languages. It presents an introduction to the concepts behind compilers and run-time representations of programming languages; features of programming languages supporting abstraction and polymorphism; and the procedural, functional, object-oriented, and concurrent programming paradigms. Programs will be required in languages illustrating each of these paradigms.

Class Format: There is no scheduled time for lectures. They will be available online for asynchronous viewing. Each conference section will meet once per week. Students should sign up for lecture and one conference.

Requirements/Evaluation: weekly problem sets and programming assignments, a midterm examination, and a final examination

Prerequisites: CSCI 136

Enrollment Limit: 20(7/conf)

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 20

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This course include regular and substantial problem sets and labs in which quantitative/formal reasoning skills are practiced and evaluated.

Fall 2020

LEC Section: R1 TBA Stephen N. Freund

CON Section: R2 MW 10:00 am - 11:15 am Stephen N. Freund

CON Section: R3 MR 1:30 pm - 2:45 pm Stephen N. Freund

CON Section: R4 MR 3:15 pm - 4:30 pm Stephen N. Freund

CON Section: R5 TR 9:45 am - 11:00 am Stephen N. Freund

Spring 2021

LEC Section: R1 ASYN Stephen N. Freund

CON Section: R2 M 10:00 am - 11:15 am Stephen N. Freund

CON Section: R3 M 1:30 pm - 2:45 pm Stephen N. Freund

CON Section: R4 T 9:45 am - 11:00 am Stephen N. Freund

CSCI 336 (F) Computer Networks (QFR)

This course explores the design and implementation of computer networks. Topics include wired and wireless networks; techniques for efficient and reliable encoding and transmission of data; addressing schemes and routing mechanisms; resource allocation for bandwidth sharing; and security issues. An important unifying themes is the distributed nature of all network problems. We will examine the ways in which these issues are addressed by current protocols such as TCP/IP and 802.11 WIFI.

Class Format: groups of three or four

Requirements/Evaluation: problem sets, programming assignments, and midterm and final examinations

Prerequisites: CSCI 136 and 237

Enrollment Limit: 18

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 18

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 337 (S) Digital Design and Modern Architecture (QFR)

This tutorial course considers topics in the low-level design of modern architectures. Course meetings will review problems of designing effective architectures including instruction-level parallelism, branch-prediction, caching strategies, and advanced ALU design. Readings will be taken from recent technical literature. Labs will focus on the development of custom CMOS circuits to implement projects from gates to bit-sliced ALUs. Final group projects will develop custom logic demonstrating concepts learned in course meetings.

Class Format: groups of three or four

Requirements/Evaluation: microprocessor design projects, participation in tutorial meetings, and examinations

Prerequisites: CSCI 237

Enrollment Limit: 18

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 18

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 338 (F) Parallel Processing (QFR)

This course explores different parallel programming paradigms used for writing applications on today's parallel computer systems. The course will introduce concurrency (i.e. multiple simultaneous computations) and the synchronization primitives that allow for the creation of correct concurrent applications. It will examine how a variety of systems organize parallel processing resources and enable users to write parallel programs for these systems. Covered programming paradigms will include multiprogramming with processes, message passing, threading in shared memory multiprocessors, vector processing, graphics processor programming, transactions, MapReduce, and other forms of programming for the cloud. Class discussion is based on assigned readings. Assignments provide students the opportunity to develop proficiency in writing software using different parallel programming paradigms.

Requirements/Evaluation: homework assignments, programming projects, and up to two exams

Prerequisites: CSCI 136 or equivalent programming experience, and CSCI 237, or permission of instructor

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will consist of substantial problem sets and programming assignments in which quantitative/formal reasoning skills are practiced and evaluated.

Not offered current academic year

CSCI 339 (S) Distributed Systems (QFR)

This course studies the key design principles of distributed systems, which are collections of independent networked computers that function as single coherent systems. Covered topics include communication protocols, processes and threads, naming, synchronization, consistency and replication, fault tolerance, and security. Students also examine some specific real-world distributed systems case studies, including Google and Amazon. Class discussion is based on readings from the textbook and research papers. The goals of this course are to understand how large-scale computational systems are built, and to provide students with the tools necessary to evaluate new technologies after the course ends.

Requirements/Evaluation: weekly homework assignments, midterm exam, 3 major programming projects, and a final project

Prerequisites: CSCI 237

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will consist of programming assignments and problem sets in which quantitative/formal reasoning skills are practiced and evaluated.

Not offered current academic year

CSCI 356 (F) Advanced Algorithms (QFR)

This course explores advanced concepts in algorithm design, algorithm analysis and data structures. Areas of focus will include algorithmic complexity, randomized and approximation algorithms, geometric algorithms, and advanced data structures. Topics will include combinatorial algorithms for packing, and covering problems, algorithms for proximity and visibility problems , linear programming algorithms, approximation schemes, hardness of approximation, search, and hashing.

Class Format: this class will follow the meeting structure of a tutorial, with groups of three or four

Requirements/Evaluation: weekly problem sets, several small programming projects, weekly paper summaries, and a small, final project

Prerequisites: CSCI 256; CSCI 361 is recommended but not required

Enrollment Limit: 10

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 10

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This class has regular and substantial problem sets in which quantitative/formal reasoning skills are practiced and evaluated.

Not offered current academic year

CSCI 357 (F) Algorithmic Game Theory (QFR)

This course focuses on topics in game theory and mechanism design from a computational perspective. We will explore questions such as: how to design algorithms that incentivize truthful behavior, that is, where the participants have no incentive to cheat? Should we let drivers selfishly minimize their commute time or let a central algorithm direct traffic? Does Arrow's impossibility result mean that all voting protocols are doomed? The overarching goal of these questions is to understand and analyze selfish behavior and whether it can or should influence system design. Students will learn how to model and reason about incentives in computational systems both theoretically and empirically. Topics include types of equilibria, efficiency of equilibria, auction design, network games, two-sided markets, incentives in computational applications such as file sharing and cryptocurrencies, and computational social choice.

Class Format: Synchronous in-class lectures will be broadcast live to remote students via zoom and recorded for asynchronous viewing. Lecture content may additionally be supplemented with prerecorded videos, and scheduled class time used as exercise or review sessions.

Requirements/Evaluation: weekly problem sets and/or programming assignments, two midterm exams, and a final project.

Prerequisites: CSCI 256 or permission of instructor

Enrollment Limit: 20

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 20

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will consist problem sets and programming assignments in which quantitative/formal reasoning skills are practiced and evaluated.

Fall 2020

LEC Section: H1 TR 11:30 am - 12:45 pm Shikha Singh

CSCI 358 (S) Applied Algorithms (QFR)

This course is about bridging the gap between theoretical running time and writing fast code in practice. The course is divided into two basic topics. The first is algorithmic: we will discuss some of the most useful tools in a coder's toolkit. This includes topics like randomization (hashing, filters, approximate counters), linear and convex programming, similarity search, and cache-efficient algorithms. Our goal is to talk about why these efficient algorithms make seemingly difficult problems solvable in practice. The second topic is applications: we will discuss how to implement algorithms in an efficient way that takes advantage of modern hardware. Specific topics covered will include blocking, loop unrolling, pipelining, as well as strategies for performance analysis. Projects and assessments will include both basic theoretical aspects (understanding why the algorithms we discuss actually work), and practical aspects (implementing the algorithms we discuss to solve important problems, and optimizing the code so it runs as quickly as possible).

Requirements/Evaluation: a course-long project and written final exam, in addition to shorter programming assignments and problem sets

Prerequisites: CSCI 256 and CSCI 237

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will consist of programming assignments and problem sets in which quantitative/formal reasoning skills are practiced and evaluated.

Not offered current academic year

CSCI 361 (F)(S) Theory of Computation (QFR)

Cross-listings: MATH 361 CSCI 361

Primary Cross-listing

This course introduces a formal framework for investigating both the computability and complexity of problems. We study several models of computation including finite automata, regular languages, context-free grammars, and Turing machines. These models provide a mathematical basis for the study of computability theory--the examination of what problems can be solved and what problems cannot be solved--and the study of complexity theory--the examination of how efficiently problems can be solved. Topics include the halting problem and the P versus NP problem.

Class Format: Lecture content will be delivered through asynchronously viewed video modules. Conference sections meeting twice per week will be used for synchronous discussions. Students should sign up for lecture and one conference section.

Requirements/Evaluation: online multiple choice and short answer questions, weekly problem sets in groups, a research project, and a final examination

Prerequisites: CSCI 256 or both a 300-level MATH course and permission of instructor

Enrollment Limit: 40(10/con)

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 40

Grading: yes pass/fail option, no fifth course option

Distributions: (D3) (QFR)

This course is cross-listed and the prefixes carry the following divisional credit:

MATH 361 (D3) CSCI 361 (D3)

Quantative/Formal Reasoning Notes: This course include regular and substantial problem sets in which quantitative/formal reasoning skills are practiced and evaluated.

Attributes: COGS Interdepartmental Electives

Fall 2020

CON Section: 02 MR 1:30 pm - 2:45 pm Aaron M. Williams

LEC Section: R1 TBA Aaron M. Williams

CON Section: R3 TR 9:45 am - 11:00 am Aaron M. Williams

Spring 2021

CON Section: H2 TR 9:45 am - 11:00 am Aaron M. Williams

CON Section: H3 MR 1:30 pm - 2:45 pm Aaron M. Williams

LEC Section: R1 ASYN Aaron M. Williams

CON Section: R4 MW 6:45 pm - 8:00 pm Aaron M. Williams

CON Section: R5 MW 8:30 pm - 9:45 pm Aaron M. Williams

CSCI 373 (S) Artificial Intelligence (QFR)

Artificial Intelligence (AI) has become part of everyday life, but what is it, and how does it work? This course introduces theories and computational techniques that serve as a foundation for the study of artificial intelligence. Potential topics include the following: Problem solving by search, Logic, Planning, Constraint satisfaction problems, Uncertainty and probabilistic reasoning, Bayesian networks, and Automated Learning.

Requirements/Evaluation: several programming projects in the first half of the semester and a larger project spanning most of the second half of the semester; reading responses and discussion; midterm examination

Prerequisites: CSCI 136 and (CSCI 256 or permission of instructor)

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Attributes: COGS Interdepartmental Electives

Not offered current academic year

CSCI 374 (F)(S) Machine Learning (QFR)

This tutorial examines the design, implementation, and analysis of machine learning algorithms. Machine Learning is a field that derives from Artificial Intelligence, Statistics, and others, and aims to develop algorithms that will improve a system's performance. Improvement might involve acquiring new factual knowledge from data, learning to perform a new task, or learning to perform an old task more efficiently or effectively. This tutorial will cover examples of supervised learning algorithms (including Bayesian approaches, support vector machines, and neural networks -- both deep and traditional), unsupervised learning algorithms (including k-means and expectation maximization), and possibly reinforcement learning algorithms (such as Q learning and temporal difference learning). It will also introduce methods for the evaluation of learning algorithms, as well as topics in computational learning theory and ethics.

Class Format: Though this course will be offered remotely by the instructor, pairs of students on campus may choose to meet in person for their tutorial sessions. If so, a classroom will be scheduled for them by the instructor.

Requirements/Evaluation: presentations, problem sets, programming exercises, empirical analyses of algorithms, critical analysis of current literature; the final two weeks are focused on a project of the student's design.

Prerequisites: CSCI 136 and CSCI 256 or permission of instructor

Enrollment Limit: 10

Enrollment Preferences: Computer Science majors

Expected Class Size: 10

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: This course heavily relies on discrete mathematics, calculus, and elementary statistics. Students will be proving theorems, among many other mathematically oriented assignments. Additionally, they will be programming, which involves analytical and logical thinking.

Attributes: COGS Interdepartmental Electives

Fall 2020

TUT Section: HT1 TBA Andrea Danyluk

Spring 2021

TUT Section: HT1 TBA Andrea Danyluk, Anna C. Neufeld

CSCI 375 (F) Natural Language Processing (QFR)

Natural language processing is a branch of computer science that studies methods for analyzing and generating written or spoken human language. It is a rapidly developing field that has given rise to many useful applications including search engines, speech recognizers, and automated personal assistants. Potential topics include information retrieval, information extraction, question answering, and language models.

Requirements/Evaluation: exams, problem sets, and programming projects

Prerequisites: CSCI 136 and (CSCI 256 or permission of instructor)

Enrollment Limit: 24

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 376 (F) Human-Computer Interaction

Cross-listings: STS 376 CSCI 376

Primary Cross-listing

Human-Computer Interaction (HCI) principles are practiced in the design and evaluation of most software, greatly impacting the lives of anyone who uses interactive technology and other products. There are many ways to design and build applications for people, so what methods can increase the likelihood that our design is the most useful, intuitive, and enjoyable? This course provides an introduction to the field of human-computer interaction, through a user-centered approach to designing and evaluating interactive systems. HCI draws on methods from computer science, the social and cognitive sciences, and interaction design. In this course we will use these methods to: ideate and propose design problems, study existing systems and challenges, explore design opportunities and tradeoffs, evaluate and improve designs, and communicate design problems and solutions to varying audiences.

Requirements/Evaluation: course projects, in-class group work/participation, and exams

Prerequisites: CSCI 136

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3)

This course is cross-listed and the prefixes carry the following divisional credit:

STS 376 (D2) CSCI 376 (D3)

Not offered current academic year

CSCI 377 (F)(S) Human Work in Computational Systems (QFR)

Cross-listings: CSCI 377 STS 375

Primary Cross-listing

As far as we know, the technological singularity has not yet arrived. Therefore, humans remain a part of our current computation pipeline. However, the role humans play varies greatly: self-driving cars aim to have human involvement only in development and emergencies, whereas educational tools are built for constant human involvement. In this course, we broadly explore human work within computational systems through topics such as crowdsourcing, educational technology, citizen science, human computation, open-source software, micro-labor markets, and online gaming. Students should expect broad exposure to a wide variety of human computing topics and group projects on building and evaluating computational systems that use human work.

Class Format: Lectures will be held on Wednesday and Friday each week. Conference sections will each meet once per week. Students should sign up for the lecture section and one conference.

Requirements/Evaluation: Course projects, in-class group work/participation, weekly written homework assignments/readings.

Prerequisites: CSCI 136

Enrollment Limit: 20

Enrollment Preferences: Preference for current CS majors

Expected Class Size: 20

Grading: yes pass/fail option, no fifth course option

Materials/Lab Fee: $75 for purchase of software and work on crowdsourcing platforms.

Distributions: (D3) (QFR)

This course is cross-listed and the prefixes carry the following divisional credit:

CSCI 377 (D3) STS 375 (D2)

Quantative/Formal Reasoning Notes: This course includes regular homework and projects in which quantitative/formal reasoning skills are practiced and evaluated.

Fall 2020

CON Section: 04 TR 9:45 am - 11:00 am Molly Q Feldman

CON Section: 05 TR 11:30 am - 12:45 pm Molly Q Feldman

LEC Section: H1 MWF 11:45 am - 1:00 pm Molly Q Feldman

CON Section: R2 W 1:30 pm - 2:20 pm Molly Q Feldman

CON Section: R3 W 2:50 pm - 3:40 pm Molly Q Feldman

Spring 2021

LEC Section: R1 MWF 11:45 am - 1:00 pm Molly Q Feldman

CON Section: R2 R 9:45 am - 11:00 am Molly Q Feldman

CON Section: R3 R 11:30 am - 12:45 pm Molly Q Feldman

CON Section: R4 R 1:30 pm - 2:45 pm Molly Q Feldman

CON Section: R5 R 3:15 pm - 4:30 pm Molly Q Feldman

CSCI 378 (F)(S) Human Artificial Intelligence Interaction

Cross-listings: STS 378 CSCI 378

Primary Cross-listing

Artificial intelligence (AI) is already transforming society and every industry today. In order to ensure that AI serves the collective needs of humanity, we as computer scientists must guide AI so that it has a positive impact on the human experience. This course is an introduction to harnessing the power of AI so that it benefits people and communities. We will cover a number of general topics such as: agency and initiative, AI and ethics, bias and transparency, confidence and errors, human augmentation and amplification, trust and explainability, and mixed-initiative systems. We explore these topics via readings and projects across the AI spectrum, including: dialog and speech-controlled systems, computer vision, data science, recommender systems, text summarization, and UI personalization, among others.

Class Format: There is no scheduled time for lectures. They will be available for asynchronous viewing. Each conference section will meet once per week, on either Tuesday or Wednesday. Students should sign up for lecture and one conference.

Requirements/Evaluation: homework, programming assignments, group work, participation, and quizzes

Prerequisites: CSCI 136, and at least one of CSCI 237, 256, or 334

Enrollment Limit: 20(8/conf)

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 20

Grading: yes pass/fail option, no fifth course option

Distributions: (D3)

This course is cross-listed and the prefixes carry the following divisional credit:

STS 378 (D2) CSCI 378 (D3)

Fall 2020

LEC Section: R1 TBA Iris Howley

CON Section: R2 TF 1:30 pm - 2:45 pm Iris Howley

CON Section: R3 TF 3:15 pm - 4:30 pm Iris Howley

CON Section: R4 MWF 9:20 am - 10:10 am Iris Howley

CON Section: R5 MWF 10:40 am - 11:30 am Iris Howley

CON Section: R6 MWF 12:00 pm - 12:50 pm Iris Howley

Spring 2021

LEC Section: R1 ASYN Iris Howley

CON Section: R2 T 1:30 pm - 2:45 pm Iris Howley

CON Section: R3 T 3:15 pm - 4:30 pm Iris Howley

CON Section: R4 W 10:00 am - 11:15 am Iris Howley

CSCI 397 (F) Independent Reading: Computer Science

Directed independent reading in Computer Science.

Requirements/Evaluation: To be determined by supervising faculty member.

Prerequisites: permission of department

Enrollment Limit: none

Enrollment Preferences: none

Expected Class Size: NA

Grading: yes pass/fail option, yes fifth course option

Distributions: (D3)

Fall 2020

IND Section: H1 TBA Stephen N. Freund

CSCI 398 (S) Independent Reading: Computer Science

Directed independent reading in Computer Science.

Requirements/Evaluation: To be determined by supervising faculty member.

Prerequisites: permission of department

Enrollment Limit: none

Enrollment Preferences: none

Expected Class Size: NA

Grading: yes pass/fail option, yes fifth course option

Distributions: (D3)

Spring 2021

IND Section: R1 TBA Stephen N. Freund

CSCI 432 (S) Operating Systems (QFR)

This course explores the design and implementation of computer operating systems. Topics include historical aspects of operating systems development, systems programming, process scheduling, synchronization of concurrent processes, virtual machines, memory management and virtual memory, I/O and file systems, system security, os/architecture interaction, and distributed operating systems.

Requirements/Evaluation: several implementation projects that will include significant programming, as well as written homework, and up to two exams

Prerequisites: CSCI 237 and either CSCI 256 or 334

Enrollment Limit: 24

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 24

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Quantative/Formal Reasoning Notes: The course will consist of substantial problem sets and/or programming assignments in which quantitative/formal reasoning skills are practiced and evaluated.

Not offered current academic year

CSCI 434 (S) Compiler Design (QFR)

This tutorial covers the principles and practices for the design and implementation of compilers and interpreters. Topics include all stages of the compilation and execution process: lexical analysis; parsing; symbol tables; type systems; scope; semantic analysis; intermediate representations; run-time environments and interpreters; code generation; program analysis and optimization; and garbage collection. The course covers both the theoretical and practical implications of these topics. Students will construct a full compiler for a simple object-oriented language.

Class Format: groups of three or four

Requirements/Evaluation: presentations, problem sets, a substantial implementation project, and two exams

Prerequisites: CSCI 237 and 256 CSCI 334 is recommended, but not required

Enrollment Limit: 10

Enrollment Preferences: current or expected Computer Science majors

Expected Class Size: 10

Grading: no pass/fail option, no fifth course option

Distributions: (D3) (QFR)

Not offered current academic year

CSCI 493 (F) Research in Computer Science

This course provides highly-motivated students an opportunity to work independently with faculty on research topics chosen by individual faculty. Students are generally expected to perform a literature review, identify areas of potential contribution, and explore extensions to existing results. The course culminates in a concise, well-written report describing a problem, its background history, any independent results achieved, and directions for future research.

Requirements/Evaluation: class participation, presentations, and the final written report

Prerequisites: none

Enrollment Limit: none

Enrollment Preferences: open to senior Computer Science majors with permission of instructor

Expected Class Size: NA

Grading: yes pass/fail option, yes fifth course option

Unit Notes: this course (along with CSCI 31 and CSCI 494) is required for students pursuing honors, but enrollment is not limited to students pursuing honors

Distributions: (D3)

Fall 2020

HON Section: H1 TBA Stephen N. Freund

HON Section: H6 TBA Daniel W. Barowy

CSCI 494 (S) Senior Thesis: Computer Science

Computer Science thesis; this is part of a full-year thesis (493-494).

Requirements/Evaluation: class participation, presentations, and the final written report

Prerequisites: CSCI 493

Enrollment Limit: none

Enrollment Preferences: open to senior Computer Science majors with permission of instructor

Expected Class Size: NA

Grading: yes pass/fail option, yes fifth course option

Distributions: (D3)

Spring 2021

HON Section: R1 TBA Stephen N. Freund

CSCI 497 (F) Independent Reading: Computer Science

Directed independent reading in Computer Science.

Requirements/Evaluation: To be determined by supervising faculty member.

Prerequisites: permission of department

Enrollment Limit: none

Enrollment Preferences: none

Expected Class Size: NA

Grading: yes pass/fail option, yes fifth course option

Distributions: (D3)

Fall 2020

IND Section: H1   TBA   Stephen N. Freund

CSCI 498 (S) Independent Reading: Computer Science

Directed independent reading in Computer Science.

Requirements/Evaluation: To be determined by supervising faculty member.

Prerequisites: permission of department

Enrollment Limit: none

Enrollment Preferences: none

Expected Class Size: NA

Grading: yes pass/fail option,   yes fifth course option

Distributions: (D3)

Spring 2021

IND Section: R1   TBA   Stephen N. Freund

Winter Study ---

CSCI 23 (W) Introduction to Research and Development in Computing

An independent project is completed in collaboration with a member of the Computer Science Department. The projects undertaken will either involve the exploration of a research topic related to the faculty member's work or the implementation of a software system that will extend the students design and implementation skills. It is expected that the student will spend 20 hours per week working on the project. At the completion of the project, each student will submit a 10-page written report or the software developed together with appropriate documentation of its behavior and design. In addition, students will be expected to give a short presentation or demonstration of their work. *Prior to the beginning of the Winter Study registration period, any student interested in enrolling must have arranged with a faculty member in the department to serve as their supervisor for the course.*

Class Format: TBA individually arranged

Requirements/Evaluation: final paper and presentation/demonstration

Prerequisites: project must be preapproved by the faculty supervisor

Enrollment Limit: POI

Enrollment Preferences: preference given to sophomores and juniors

Grading: pass/fail only

Materials/Lab Fee: none

Not offered current academic year

CSCI 31 (W) Senior Thesis: Computer Science

To be taken by students registered for Computer Science 493-494.

Class Format: independent study

Grading: pass/fail only

Not offered current academic year

CSCI 99 (W) Independent Study: Computer Science

Open to upperclass students. Students interested in doing an independent project (99) during Winter Study must make prior arrangements with a faculty sponsor. The student and professor then complete the independent study proposal form available online. The deadline is typically in late September. Proposals are reviewed by the pertinent department and the Winter Study Committee. Students will be notified if their proposal is approved prior to the Winter Study registration period.

Class Format: independent study

Grading: pass/fail only

Not offered current academic year